Universidade Católica de Brasília

PRÓ-REITORIA DE PÓS-GRADUAÇÃO E PESQUISA

MESTRADO EM GESTÃO DO CONHECIMENTO E DA TECNOLOGIA DA INFORMAÇÃO

BLOOM – BLAST Object Oriented Management: uma solução integrada para gerenciamento dos resultados do BLAST por meio de um paradigma orientado a objetos.

Leila de Fátima Sousa Carvalho

Brasília-DF
2002

E R R A T A

Página	*Linha*	*Onde se lê*	*Leia-se*

Leila de Fátima Sousa Carvalho

BLOOM – BLAST Object Oriented Management:
uma solução integrada para gerenciamento dos resultados do BLAST por meio de um paradigma orientado a objetos.

"DISSERTAÇÃO/TESE APRESENTADA AO PROGRAMA DE PÓS-GRADUAÇÃO STRICTO SENSU EM GESTÃO DO CONHECIMENTO E DA TECNOLOGIA DA INFORMAÇÃO DA UNIVERSIDADE CATÓLICA DE BRASÍLIA, COMO REQUISITO PARCIAL PARA OBTENÇÃO DO GRAU DE MESTRE EM GESTÃO DO CONHECIMENTO E DA TECNOLOGIA DA INFORMAÇÃO."

Orientadores: Prof. Dr. **Marcos Mota do Carmo Costa** e
Prof. Dr. **Georgios Joannis Pappas Júnior**.

Brasília-DF
2002

C331B Carvalho, Leila de Fátima Sousa

BLOOM-BLAST Object Oriented Management: uma solução integrada para gerenciamento dos resultados do BLAST por meio de um paradigma orientado a objetos / Leila de Fátima Sousa Carvalho. – Brasília, 2002.

203 f. : il

Orientadores: Marcos Mota do Carmo Costa e Georgios Joannis Pappas Júnior.

Dissertação (mestrado) – Universidade Católica de Brasília, 2002.

1. Biologia computacional - bioinformática. 2. Análise genômica. 3. Biotecnologia. I. Costa, Marcos Mota do Carmo, orient. II. Pappas Júnior, Georgios Joannis, orient. III. Título.

CDU 57.081.2

Ficha elaborada pela Divisão de Processamento do Acervo do SIBI – UCB.

Dissertação defendida e aprovada, em 27 de março de 2003, pela banca examinadora constituída pelos professores:

Presidente e Orientador - Prof. Marcos Mota do Carmo Costa, Dr. - UCB

Prof. Rogério Alvarenga, Dr. – UCB

Prof. Wellington Martins, Dr. – UCB

Prof[a.] Natália Florêncio Martins, Dra. – Embrapa/Cenargen

Esta Dissertação é dedicada a Deus, que me proporcionou saúde e coragem para cumprir mais esta etapa de vida, à minha família e aos amigos pelo carinho e compreensão.

AGRADECIMENTOS

A conclusão desta Dissertação só se tornou possível graças ao trabalho de diversos colaboradores. Agradecemos a todos e, de forma particular:

aos professores Dr. Marcos Mota do Carmo Costa, Dr. Georgios Pappas e Dr. Wellington Martins pela indicação desta excelente linha de trabalho, pela inestimável orientação e companheirismo;

à Profa. Dra. Natália Florêncio Martins pela gentileza, disponibilidade e paciência em ajudar-me a corretamente compreender e aplicar as noções de Biologia Molecular;

aos pesquisadores e estagiários do Laboratório de Bioinformática do Cenargen, especialmente Dr. Felipe Rodrigues da Silva e David Fagundes Junior, respectivamente;

à empresa EMBRAPA e aos colegas de trabalho pela oportunidade e apoio, especialmente Antenor Turazi e Adelina Sesconetto Borges;

a todos meus professores do Mestrado pela contribuição significativa à realização deste trabalho;

aos funcionários da pós-graduação da Universidade Católica de Brasília, especialmente Leonor Gonçalves e Mônica Miranda, pela atenção;

à amiga e professora de inglês Melissa Dias Hecksher pelo carinho, incentivo e por sua inestimável colaboração na revisão deste trabalho e na correção das traduções de lingua inglesa.

aos amigos e professores Fernando Monteiro Ribeiro (UNB) e Rodrigo Bonifácio Almeida (UCB) pela gentileza, disponibilidade e indispensável contribuição técnica na condução desta Pesquisa.

às amigas Patrícia Cyriaco da Silva e Wanderlea Rodrigues de Souza e aos meus alunos, que me incentivaram e colaboraram fortemente para a conclusão deste trabalho;

à minha família, especialmente meus pais, pelas dedicadas orações e pelo jeito simples de manifestarem seu orgulho por mais esta conquista em nossas vidas.

Analisar a vida é como observar num caleidoscópio: os mesmos elementos, mas, a cada movimento, uma imagem diferente.

(Armando Oscar Cavanha)

SUMÁRIO

LISTA DE FIGURAS VIII
LISTA DE TABELAS X
LISTA DE ABREVIATURAS E SÍMBOLOS MAIS UTILIZADOS XI
RESUMO XII
ABSTRACT XIII
1. INTRODUÇÃO 1
1.1 ORGANIZAÇÃO DO TRABALHO 6
2. OBJETO DA PESQUISA 7
2.1 MOTIVAÇÃO 7
2.2 OBJETIVOS OU PROPOSIÇÕES 12
2.2.1 Objetivo geral 12
2.2.2 Objetivos específicos 13
3. REFERENCIAL TEÓRICO DE BIOLOGIA MOLECULAR COMPUTACIONAL 15
3.1 NOÇÕES BÁSICAS DE BIOLOGIA MOLECULAR 15
3.1.1 A célula 15
3.1.2 Ácidos nucléicos 16
I. DNA 16
II. RNA 20
3.1.3 Proteínas 21
I. Aminoácidos 21
II. Proteínas: definição, funções e estrutura 24
3.1.4 O Dogma Central revisado 28
3.2 COMPARAÇÃO DE SEQÜÊNCIAS E BUSCA EM BANCO DE DADOS 33
3.2.1 Tipos de comparação e a importância da busca em bancos de dados 33
3.2.2 Similaridade e Homologia 37
3.2.3 Tipos de Homologia 40
3.3 MÉTODOS DE ALINHAMENTO 41
3.3.1 Medidas de distância e similaridade 41
3.3.2 Principais métodos de alinhamento e alinhamento *pairwise* 45
3.3.3 A importância evolutiva dos *gaps* e suas funções de penalidade 51
3.3.4 Algoritmos 53
3.3.5 Algoritmos de Programação Dinâmica 53
3.3.6 Matrizes de substituição ou de escore 65
3.3.7 PAM 66
3.3.8 BLOSUM 70
I. O banco de dados PROSITE 70
II. O banco de dados BLOCKS 73
III. As matrizes BLOSUM 73
3.3.9 Algoritmos heurísticos para buscas em bancos de dados 77
I. FASTA 78
II. BLAST 86
III. Comparação entre BLAST e FASTA 94
3.4 ENTENDENDO UM PROJETO GENOMA 95
3.5 ALINHAMENTO MÚLTIPLO 103
3.6 PREDIÇÃO DE ESTRUTURA SECUNDÁRIA DE PROTEÍNA 110
3.7 DERIVAÇÃO DE ÁRVORE FILOGENÉTICA 114
4. METODOLOGIA 122
4.1 MODELAGEM ORIENTADA A OBJETOS 122
4.2 RUP 124
4.3 JAVA 128
4.4 *DESIGN PATTERNS* E ORIENTAÇÃO A OBJETOS 129
4.4.1 Os *patterns* mais utilizados no Sistema proposto 132
4.5 CORBA 135
4.6 XML 139
5. VISÃO GERAL DO PROJETO 141
5.1 PROPRIEDADE INTELECTUAL 141

5.2 PLANO DE DESENVOLVIMENTO 141
5.3 OPORTUNIDADE CIENTÍFICA 142
5.4 DETALHAMENTO OU SENTENÇA DO PROBLEMA 142
5.5 POSICIONAMENTO DO PRODUTO 145
5.6 DESCRIÇÃO DA EQUIPE CLIENTE E DOS USUÁRIOS 146
5.7 INTERAÇÕES OU PERSPECTIVAS DO PRODUTO 146
5.8 APLICABILIDADE DE PADRÕES 151
5.9 LISTA DE RISCOS 152
5.10 ARQUITETURA CANDIDATA 153
5.10.1 Organização do sistema em camadas 153
5.10.2 Mecanismo de persistência 154
5.10.3 Comunicação com sistemas externos 154
5.11 VISÃO DE CASOS DE USO 155
5.11.1 Atores 155
5.11.2 Casos de uso do ator Pesquisador 157
5.11.3 Casos de uso do ator Clock 158
5.11.4 Prioridade dos casos de uso 159
5.11.5 Detalhamento de alguns casos de uso 160
I. Visualizar alinhamento *pairwise* 160
II. Construir alinhamento múltiplo 163
III. Predizer estrutura secundária de proteína 165
IV. Derivar árvore filogenética 167
5.12 DISTRIBUIÇÃO DAS ITERAÇÕES 170
5.13 VISÃO DE PROJETO 172
5.13.1 Diagramas de classes 172
5.13.2 Diagramas de interação 174
I. Diagramas de seqüência 174
6. RESULTADOS 180
7. CONCLUSÃO 182
8. DESENVOLVIMENTOS FUTUROS 183
9. REFERÊNCIAS BIBLIOGRÁFICAS 185
ANEXO 190
GLOSSÁRIO 198

LISTA DE FIGURAS

Figura 1.1-1 – Crescimento do GenBank. 1
Figura 1.1-2 – Crescimento do PDB....2
Figura 1.1-3 – Exemplo de execução do BLAST para uma seqüência de nucleotídeos. 4
Figura 1.1-4 – Exemplo de resultado da consulta BLAST para uma seqüência de nucleotídeos. 5
Figura 3.1-1 – A dupla hélice do DNA....17
Figura 3.1-2 – A estrutura do cromossomo. 18
Figura 3.1-3 – RNA e DNA....20
Figura 3.1-4 – Estrutura química do aminoácido....21
Figura 3.1-5 – Os 4 níveis estruturais de uma proteína. 26
Figura 3.1-6 – Lado esquerdo: estrutura terciária da hexoquinase. Lado direito: estrutura quaternária da hemoglobina (com quatro polipeptídeos, sendo dois alfa-globina e dois beta-globina). 27
Figura 3.1-7 – Lado esquerdo: cadeia B da Protein Kinase C Interacting Protein e corresponde à estrutura em azul no outro lado. Lado direito: estrutura quaternária da mesma proteína....27
Figura 0-1 – O Dogma Central revisado da Biologia Molecular. 30
Figura 0-2 – A Síntese de Proteína. 32
Figura 3.2-1 – Árvore da vida....37
Figura 3.2-2 – Esquema simplificado de possível evolução de duas espécies a partir de um ancestral comum....38
Figura 3.2-3 – Um exemplo de homologia de seqüências. 41
Figura 3.3-1 – A fórmula da relação recorrente de programação dinâmica. 56
Figura 3.3-2 – A condição-base da relação recorrente do alinhamento global para as seqüências s=AAAC e t=AGC. ..56
Figura 3.3-3 – A condição-base da relação recorrente do alinhamento local para as seqüências s=AAAC e t=AGC....56
Figura 3.3-4 – Um algoritmo (pseudocódigo) para calcular a similaridade global em programação dinâmica....57
Figura 3.3-5 – Um algoritmo (pseudocódigo) para calcular a similaridade local em programação dinâmica. 58
Figura 3.3-6 – O início da computação tabular para alinhamento global das seqüências s=AAAC e t=AGC (parte 1 de 2). 59
Figura 3.3-7 – O resultado final da computação tabular para alinhamento global das seqüências s=AAAC e t=AGC (parte 2 de 2). 59
Figura 3.3-8 – O início da computação tabular para alinhamento local das seqüências s=AAAC e t=AGC 60
Figura 3.3-9 – O resultado final da computação tabular para alinhamento local das seqüências s=AAAC e t=AGC (parte 2 / 2). 60
Figura 3.3-10 – Um exemplo de algoritmo (pseudocódigo) para calcular o alinhamento ótimo em programação dinâmica. 61
Figura 3.3-11 – Preferência anti-horária das setas no rastreamento reverso. 62
Figura 3.3-12 – O alinhamento global ótimo em programação dinâmica para as seqüências s=AAAC e t=AGC. 63
Figura 3.3-13 – O alinhamento local ótimo em programação dinâmica para as seqüências s=AAAC e t=AGC. 64
Figura 3.3-14 – Comparação do grau de divergência entre as matrizes BLOSUM e PAM....75
Figura 3.3-15 – Primeira etapa do FASTA : localização dos “hot spots” (k-tuplas). 81
Figura 3.3-16 – Segunda etapa do FASTA : método da diagonal. 83
Figura 3.3-17 – Terceira etapa do FASTA : junção das regiões iniciais de diagonais diferentes. 84
Figura 3.3-18 – Quarta etapa do FASTA : determinação do alinhamento ótimo. 85
Figura 3.3-19 – BLAST: *Compilação de uma lista com palavras de alto escore (“semente”).* 90
Figura 3.3-20 – BLAST: *Tabela de busca e comparação das seqüências do BD com as entradas desta tabela*. 92
Figura 3.3-21 – BLAST: Autômato finito determinístico para reconhecer a palavras vizinhas QL, QM e ZL. 92
Figura 3.3-22 – BLAST: *Extensão das sementes de cada uma das seqüências*. 93
Figura 3.4-1 – Seqüenciamento por clonagem hierárquica e por shotgun. 99
Figura 3.5-1 – Exemplo de alinhamento múltiplo com seqüência consenso. 104
Figura 3.5-2 – Exemplo de alinhamento local e global. 107
Figura 3.5-3 – O processo de alinhamento progressivo. 109
Figura 3.6-1 – A estrutura tridimensional da proteína Flavodoxina. 113
Figura 3.6-2 – Exemplo de predição de estrutura secundária de proteína. (parte 1 / 2). 113
Figura 3.6-3 – Exemplo de predição de estrutura secundária de proteína. (parte 2 / 2). 114
Figura 3.7-1 – Esquema de uma árvore filogenética. 117
Figura 3.7-2 – Exemplo de cladograma e aninhamento de parênteses correspondente. 118
Figura 3.7-3 – Quadro explicativo dos métodos utilizados na construção de árvores filogenéticas moleculares....118
Figura 3.7-4 – Exemplo de árvore sem raiz e árvore UPGMA correspondente. 119
Figura 4.2-1 – Ciclo de vida do RUP....126

Figura 4.4-1 – Esquema do *pattern* Factory. 134
Figura 4.4-2 – Esquema do *pattern* Facade. 135
Figura 4.6-1 – Fluxo de dados na ferramenta Castor. 140
Figura 5.7-1 – Esquema das interações do Sistema. 150
Figura 5.10-1 – Organização do Sistema em camadas. 153
Figura 5.10-2 – Diagrama de classes do mecanismo de persistência do Sistema. 154
Figura 5.11-1 – Diagrama dos Casos de Uso do ator Pesquisador 158
Figura 5.11-2 – Diagrama dos Casos de Uso do ator Clock 159
Figura 5.11-3 – Diagrama de Atividades – Visualizar alinhamento *pairwise.* 162
Figura 5.11-4 – Diagrama de Atividades – Construir alinhamento múltiplo. 165
Figura 5.11-5 – Diagrama de Atividades – Predizer estrutura secundária de proteína. 167
Figura 5.11-6 – Diagrama de Atividades – Derivar árvore filogenética. 169
Figura 5.13-1 – Sistema BLOOM: diagrama de classes dos serviços parametrização e filtragem. 172
Figura 5.13-2 – Sistemas Genoma e BLOOM: diagrama de classes de resultado do BLAST (comuns a ambos). 173
Figura 0-1 – Sistema Genoma: diagrama de seqüência do cenário Inicializa. 174
Figura 0-2 – Sistema Genoma: diagrama de seqüência do cenário Resultados do BLAST. 175
Figura 0-1 – Sistema BLOOM: diagrama de seqüência do cenário Visualizar alinhamento *pairwise*. 176
Figura 0-1 – Sistema BLOOM: diagrama de seqüência do cenário Construir alinhamento múltiplo. 177
Figura 0-1 – Sistema BLOOM: diagrama de seqüência do cenário Predizer estrutura secundária de proteína. 178
Figura 0-2 – Sistema BLOOM: diagrama de seqüência do cenário Derivar árvore filogenética. 179
Figura 0-1 – Um exemplo de resultado BLAST no formato XML. 192
Figura 0-2 – Sistemas Genoma e BLOOM: diagrama de classes de controle comuns a ambos. 193
Figura 0-1 – Sistema Genoma: validação do usuário e entrada. 194
Figura 0-2 - Sistema Genoma: módulo de Consulta – Seleção da placa 194
Figura 0-3 - Sistema Genoma: módulo de Análise de Seqüência. 195
Figura 0-4 – Sistema BLOOM: visualização gráfica do resultado do BLAST. 196
Figura 0-1 – Sistema BLOOM: sumário do resultado do BLAST 197

LISTA DE TABELAS

Tabela 3.1-1 – Os vinte aminoácidos comumente encontrados nas proteínas. 22
Tabela 3.1-2 – O código genético mapeando códons para aminoácidos. 24
Tabela 3.3-1 – Distância Levenshtein ou de edição para transformar “LEITOR” em “ESCRITOR”. 43
Tabela 3.3-2 – Distância Levenshtein ou de edição para transformar “AGCACACA” em “ACACACTA”. 43
Tabela 3.3-3 – Distância Hamming entre as seqüências *s* e *t*. 44
Tabela 3.3-4 – Alinhamento global entre as seqüências s=LEITOR a t=ESCRITOR. 46
Tabela 3.3-5 – Alinhamento global entre as seqüências s=CGGATTAC a t=CGGATTCA. 47
Tabela 3.3-6 – Alinhamento local entre as seqüências *s*=PQRAXABCSTVQ e *t*=XYAXBACSLL. 48
Tabela 3.3-7 – Alinhamento semiglobal (1) entre as seqüências *s*=CAGCACTTGGATTCTCGG e *t*=CAGCGTGG. 49
Tabela 3.3-8 – Alinhamento semiglobal (2) entre as seqüências *s*=CAGCACTTGGATTCTCGG e *t*=CAGCGTGG. 49
Tabela 3.3-9 – Um exemplo de alinhamento *pairwise* mostrando duas lacunas e dois espaços isolados. 51
Tabela 3.3-10 – Alinhamento global ótimo para as seqüências *s*=AAAC e *t*=AGC. 62
Tabela 3.3-11 – Alinhamento global ótimo para as seqüências *s*=AAAC e *t*=AGC. 63
Tabela 3.3-12 – A matriz PAM250. 68
Tabela 3.3-13 – A matriz BLOSUM62. 76
Tabela 3.3-14 – Os algoritmos da família FAST. 79
Tabela 3.3-15 – Os algoritmos da família BLAST. 87
Tabela 3.3-16 – Um par de segmentos com escore determinado pela matriz BLOSUM62. 88
Tabela 3.3-17 – BLAST: Seqüência de busca, parâmetros e palavras com alto escore (matriz BLOSUM62). 91
Tabela 3.6-1 – As 3 gerações de enfoques para predição de estrutura secundária de proteína. 112
Tabela 5.2-1 – Plano de desenvolvimento. 141
Tabela 5.4-1 – Detalhamento ou sentença do problema. 145
Tabela 5.5-1 – Posicionamento do produto. 145
Tabela 5.6-1 – Equipe cliente. 146
Tabela 5.6-2 – Potenciais usuários do Sistema. 146
Tabela 5.9-1 – Lista de riscos. 152
Tabela 5.11-1 – Prioridade dos casos de uso. 159
Tabela 5.12-1 – Distribuição das iterações. 171

LISTA DE ABREVIATURAS E SÍMBOLOS MAIS UTILIZADOS

BD	- Banco de Dados
Biofoco	- Rede de Bioinformática do Centro-Oeste
BLAST	- *Basic Local Alignment Search Tool* ou Ferramenta Básica de Busca de Alinhamento Local
BLOSUM	- *Block Substitution Matrices* ou Matrizes de Substituição de Blocos
bp	- *Base pair*, par de bases ou simplesmente bases
cDNA	- DNA complementar
CENARGEN	- Embrapa Recursos Genéticos e Biotecnologia
CORBA	- *Common Object Request Broker Architecture* ou Arquitetura Comum de Agente de Requisição de Objeto
DNA	- *DeoxyriboNucleic Acid* ou Ácido Desoxirribonucléico
DTD	- *Document Type Definition* ou Definição de Tipo de Documento
EMBRAPA	- Empresa Brasileira de Pesquisa Agropecuária
ESTs	- *Expressed Sequence Tags* ou Etiquetas de Seqüências Expressas
FASTA	- *Fast Alignment* ou Alinhamento Rápido
HTML	- *HyperText Markup Language* ou Linguagem de Marcação de Hipertexto
J2EE	- *Java2 Enterprise Edition* ou Edição Corporativa de Java2.
mRNA	- RNA mensageiro
NCBI	- *National Center for Biotechnology Information* ou Centro Nacional de Informação Biotecnológica
OGM	- Organismo Geneticamente Modificado
OMG	- *Object Management Group* ou Grupo de Gerenciamento de Objetos
ORESTES	- *Open Reading Frames ESTs* ou ESTs de Região de Leitura Aberta
ORF	- *Open Reading Frame* ou Região de Leitura Aberta
PAM	- *Point Accepted Mutation* ou Mutação Pontual Aceita
PCR	- *Polimerase Chain Reaction* ou Reação em Cadeia da Polimerase
PDB	- *Protein Data Bank* ou Banco de Dados de Proteína
PDS	- *Persistent Data Service* ou Serviço de Dados Persistentes
PHRAP	- *PHRagment Assembly Program (ou phil's revised assembly program*) ou programa de montagem de fragmentos
POS	- *Persistent Object Service* ou Serviço de Objeto Persistente
PSI-BLAST	- *Position Specific Iterative BLAST* ou BLAST Iterativo de Posição Específica
RNA	- *RiboNucleic Acid* ou Ácido Ribonucléico
rRNA	- RNA ribossômico
RUP	- *Rational Unified Process* ou Processo Rational Unificado
SQL	- *Structure Query Language* ou Linguagem Estruturada de Consulta
tRNA	- RNA transportador
UCB	- Universidade Católica de Brasília
UML	- Unified Modeling Language ou Linguagem Unificada de Modelagem
UNB	- Universidade de Brasília
XML	- *eXtended Markup Language* ou Linguagem Extensível de Marcações

RESUMO

Considerada uma disciplina especial desde o início dos anos 80, a bioinformática pode ser definida como uma modalidade que abrange todos os aspectos de aquisição, processamento, armazenamento, distribuição, análise e interpretação da informação biológica. Tudo ocorre numa estreita sinergia com o paradigma fundamental da biologia molecular, a qual postula que a informação genética está armazenada nas seqüências de DNA.

Após a iniciativa pública do Projeto do Genoma Humano, iniciado em 1990 e com prazo previsto para ser completado em 2005, tem havido crescimento extraordinário do volume de seqüências de nucleotídeos e de aminoácidos disponível em bancos de dados. Para que se converta em efetivo benefício na medicina, na biotecnologia, na agronomia e em diversas outras áreas, toda essa informação genética precisa ser processada, comparada e analisada, o que constitui novos desafios para bioinformática, matemática e estatística: integrar e disponibilizar esses dados por meio de ferramentas amigáveis que permitam a mineração eficiente de informações pelos especialistas, mesmo aqueles pouco afeitos a computadores e a biologia molecular.

A proposta desta pesquisa é integrar o maior número possível de serviços de análise de seqüências, encapsulando-os em um único aplicativo de interface gráfica interativa. Inicialmente estão sendo considerados quatro serviços, a saber: alinhamento *pairwise*, alinhamento múltiplo, predição de estrutura secundária de proteína e derivação de árvore filogenética. Com isso, pretende-se tornar a análise genômica mais acessível, eficiente e simplificada, o que permitirá aos pesquisadores concentrarem-se principalmente na interpretação biológica dos resultados.

Quanto à bioinformática, os resultados obtidos nesta pesquisa permitem concluir que o aplicativo em desenvolvimento vem possibilitando aos pesquisadores da Rede Biofoco: melhor gerenciamento, organização e disponibilização das informações genômicas; e uso desse como ferramenta de *datamining*, auxiliando na visualização e exploração das informações que trafegam como entrada ou saída entre as diversas ferramentas integradas. Quanto à informática, alguns dos principais ganhos são: validação e emprego de orientação a objetos, CORBA, UML e RUP no desenvolvimento de uma ferramenta para uso científico; integração com utilitários externos e com os sistemas Anotação e Genoma; e reuso da arquitetura empregada no sistema Genoma.

Palavras-chaves

Bioinformática, Biologia molecular computacional, DNA, RNA, Proteína, Comparação de seqüências, Métodos de alinhamento de seqüências, Programação dinâmica, PAM, BLOSUM, BLAST, FAST, Estrutura secundária de proteína, Árvore filogenética, Projeto genoma.

ABSTRACT

Considered a special subject since the beginning of the 80's, bioinformatics can be defined as a way that covers all aspects of acquisition, processing, storage, distribution, analysis and interpretation of biological information. All this happens in a narrow sinergy with the molecular biology fundamental paradigm, which postulates that the genetic information is stored in the DNA sequences.

After the public initiative of Human Genome Project, started in 1990 and its deadline to be completed in 2005, there has been a great deal of growth in the volume of the nucletide sequences and aminoacids available in databases. In order to be converted into an effective benefit in medicine, agronomy and many other areas, all this genetic information needs to be processed, compared and analysed, which are the new challenges of bioinformatics, mathematics and statistics: to integrate and to make these data available through tools that allow specialists to look into such information, even those who are not into computer and molecular biology.

The purpouse of this research is to integrate the largest possible number of services of sequence analysis, condensing them in one aplication of interative graphic interface. Initially four services are to be considerate: pairwise alignment, multiple alignment, protein secondary structure prediction and phylogenetic tree derivation. Through this study, it is supposed to make the genomic analysis more accessible, efficient and simplified, allowing researchers to concentrate mainly on the results of the biological interpretation.

As far as bioinformatics is concerned, the results obtained in this research leads to the conclusion that the developing application software has made possible for the Biofoco researchers: better management, organization and availability of the genomic information; and the use of this software as a datamining tool, helping the visualization and exploration of the information that travels in and out through the many integrated tools. Concerning informatics, some of its main achievements are: validation and use of object-oriented technologies, CORBA, UML and RUP in the development of a scientific purpouse tool; integration with external utilities and with the Annotation and Genoma Systems; and integral reuse of the architecture used in the Genoma System.

Keywords

Bioinformatics, Computational molecular biology, DNA, RNA, Protein, Sequence comparison, Sequence alignment methods, Dynamic programming, PAM, BLOSUM, BLAST, FAST, Protein secondary structure, Phylogenetic tree, Genome project.

1. INTRODUÇÃO

A Biologia é a mais recente das ciências naturais. Quando os dados coletados chegam a uma densidade crítica, uma ciência natural progride da simples coleta para o processamento dessas informações acumuladas. Essa última atividade tem se tornado dominante também em outras ciências já bastante maduras, tais como a Física, na qual predições e abstrações teóricas desempenham um papel importante devido à escassez de novas informações **[Kim, 2002]**.

Até recentemente, a maior atividade da Biologia vinha sendo o acúmulo de novos dados, tanto de laboratório quanto de campo. Por isso, o crescimento do volume dessas informações nos últimos cinco anos, especialmente em nível molecular, tem sido surpreendente. A curva de crescimento da informação armazenada no GenBank (http://www.ncbi.nlm.nih.gov/Genbank/genbankstats.html), um dos mais importantes bancos de dados de biologia molecular, é uma curva exponencial que reproduz perfeitamente a lei de Moore[1] – dobrar o tamanho a cada 18 meses. Apenas para efeito ilustrativo, o tamanho total do GenBank já ultrapassa 14 milhões de seqüências, apresentando um crescimento acumulado de 200% nos últimos três anos; e o arquivo de proteínas PDB também já possui um volume total acima de 17.000 diferentes seqüências de proteína, o que significa um crescimento acima de 50% desde 1999, conforme mostram os gráficos abaixo (Figuras 1.1-1 e 1.1-2):

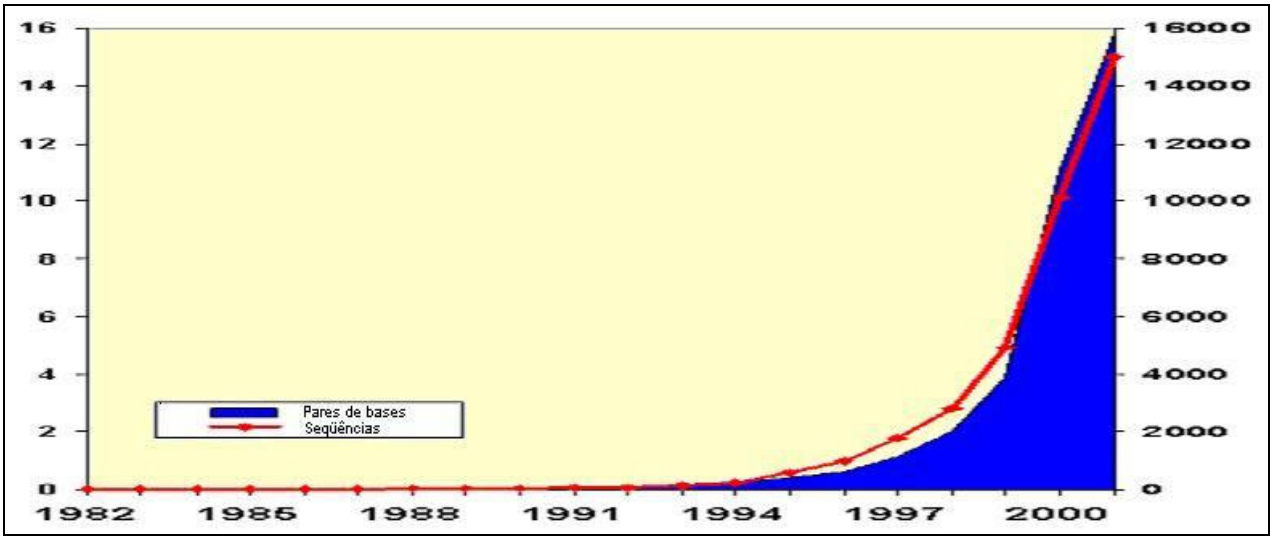

Figura 1.1-1 – Crescimento do GenBank.
Atualizado em 12/03/2002. A parte à direita e mais escura (azul) é o número de pares de bases em milhões de unidades; a linha com traços é o número de seqüências também em milhões de unidade.
Fonte http://www.ncbi.nlm.nih.gov/Genbank/genbankstats.html.

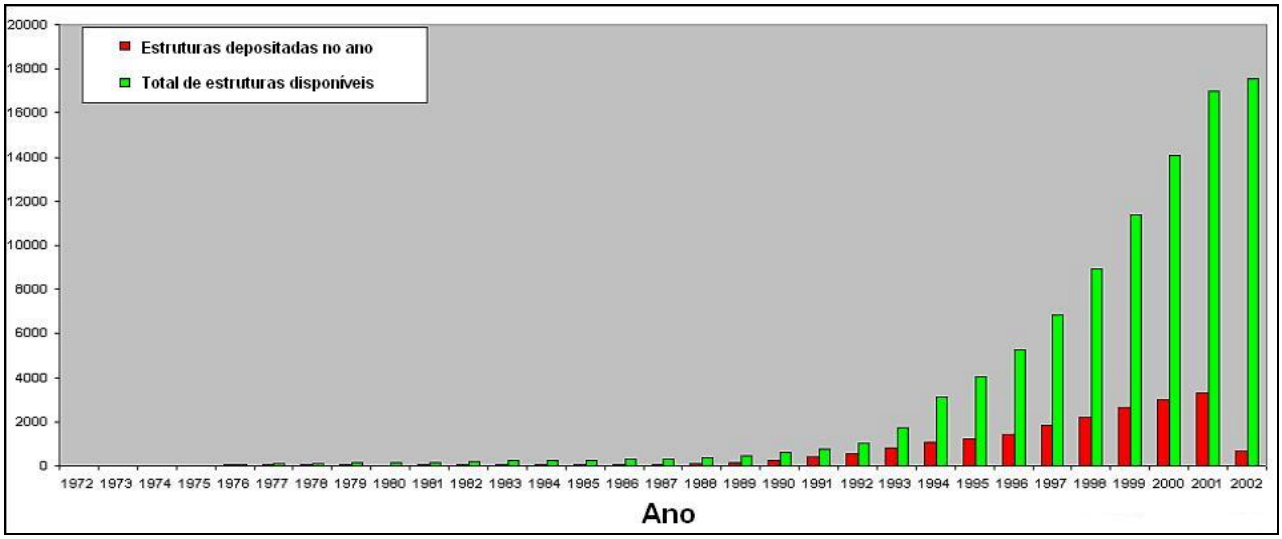

Figura 1.1-2 – Crescimento do PDB.
Atualizado em 12/03/2002. As barras menores e à esquerda são o número de seqüências depositadas por ano; as barras à direita são o número total de seqüências disponíveis. Fonte http://www.rcsb.org/pdb/holdings.html.

Conforme relato pessoal de **[Kim, 2002]**, há dez anos, para se obter a seqüência de 200 pares de bases (ou bp – *base pair*)[2] de DNA, isso levava aproximadamente 5 dias. Há dois anos, a Celera, uma corporação especializada em biotecnologia, seqüenciou o genoma completo da *Drosophila melanogaster*[3] - algo em torno de 165 milhões de pares de bases – em poucos meses, o que atualmente pode ser executado em questão de semanas. E **[Heath e Ramakrishnan, 2002]** ainda citam os casos dos genomas do arroz e da planta *Arabidopsis thaliana*[4], seqüenciados recentemente e em poucos meses.

Um outro exemplo, a iniciativa pública do Projeto Genoma Humano, iniciado em 1990 e com prazo previsto para ser completado em 2005, tinha seqüenciado até 1998 cerca de apenas 3% do genoma. Nos dois anos seguintes, com a introdução de novos seqüenciadores capilares de DNA, completou-se o trabalho restante de geração de seqüências. O consórcio público seqüenciou 22 bilhões de bp e a CELERA[i], cerca de 14.5 bilhões de bp. Estimativas atuais indicam que a capacidade de seqüenciamento do setor público dedicado apenas ao Projeto do Genoma Humano já atingiu, em média, 28 milhões de pares de bases por mês. Não é exagero afirmar que as novas tecnologias permitem hoje que um seqüenciador faça, no intervalo de poucas horas, o que grupos de seqüenciadores faziam no início da década de 90 no período de um ano **[Pereira, 2001]**.

[i] CELERA - Empresa privada pertencente majoritariamente à Applied Biosystem, uma companhia que fabrica seqüenciadores.

Tanto pelo volume quanto pela rapidez com que são produzidas, essas informações suplantam a capacidade de serem tratadas por qualquer que seja a pessoa, ou mesmo por um grupo de pessoas. **[Pappas, 2002]** Para viabilizar a extração das informações pertinentes e realmente poder concretizar as potencialidades oriundas de se possuir o genoma completo de um organismo, fortaleceu-se um ramo da ciência, surgido no início dos anos 80: a Bioinformática, uma modalidade que abrange todos os aspectos de aquisição, processamento, armazenamento, distribuição, análise e interpretação da informação biológica. Tudo ocorre numa estreita sinergia com o paradigma fundamental da biologia molecular, a qual postula que a informação genética está armazenada nas seqüências de DNA. A bioinformática é também auxiliada por diversos algoritmos e procedimentos de matemática e estatística.

Atualmente, os três usos mais bem sucedidos do computador em biologia são: modelagem de estruturas[5], análise comparativa de seqüências e clonagem *in silico*, que é o processo de usar uma busca em bancos de dados existentes para clonar um gene. No entanto, para esta Dissertação, o enfoque estará voltado apenas para o segundo uso **[Kim, 2002]**.

Na análise comparativa, quando pesquisadores decifram os pares de bases de um fragmento genômico em laboratório, surge imediatamente a pergunta: **[Pappas, 2002]** "Qual seria a função celular desta seqüência nucleotídica?" Com isso, verifica-se que a acumulação em massa de dados genômicos somente se justifica caso seja possível interpretá-los consistentemente. Um primeiro passo é verificar se outras seqüências similares e homólogas já foram estudadas, pois conhecendo a função de uma, pode-se transferir essa informação para as seqüências relacionadas. Essa verificação pode ser executada, de forma rápida, pela utilização de programas de computador. Chega-se, portanto, a um quadro em que os experimentos passam a ser realizados primeiramente *in silico* para depois serem confirmados *in vivo* ou *in vitro*.

Conforme atesta **[Kim, 2002]**, provavelmente a ferramenta computacional mais largamente usada para análise comparativa é o BLAST **[NCBI/Education, 2002],** que pesquisa bancos de dados como o GenBank em busca de todas seqüências similares a uma determinada seqüência de busca (*target ou query sequence*). Nos dias de hoje,

quando pesquisadores isolam uma nova seqüência molecular, a primeira coisa que fazem é executar o BLAST contra os bancos de dados existentes.

Segue um exemplo de chamada do BLASTn (Figura 1.1-3):

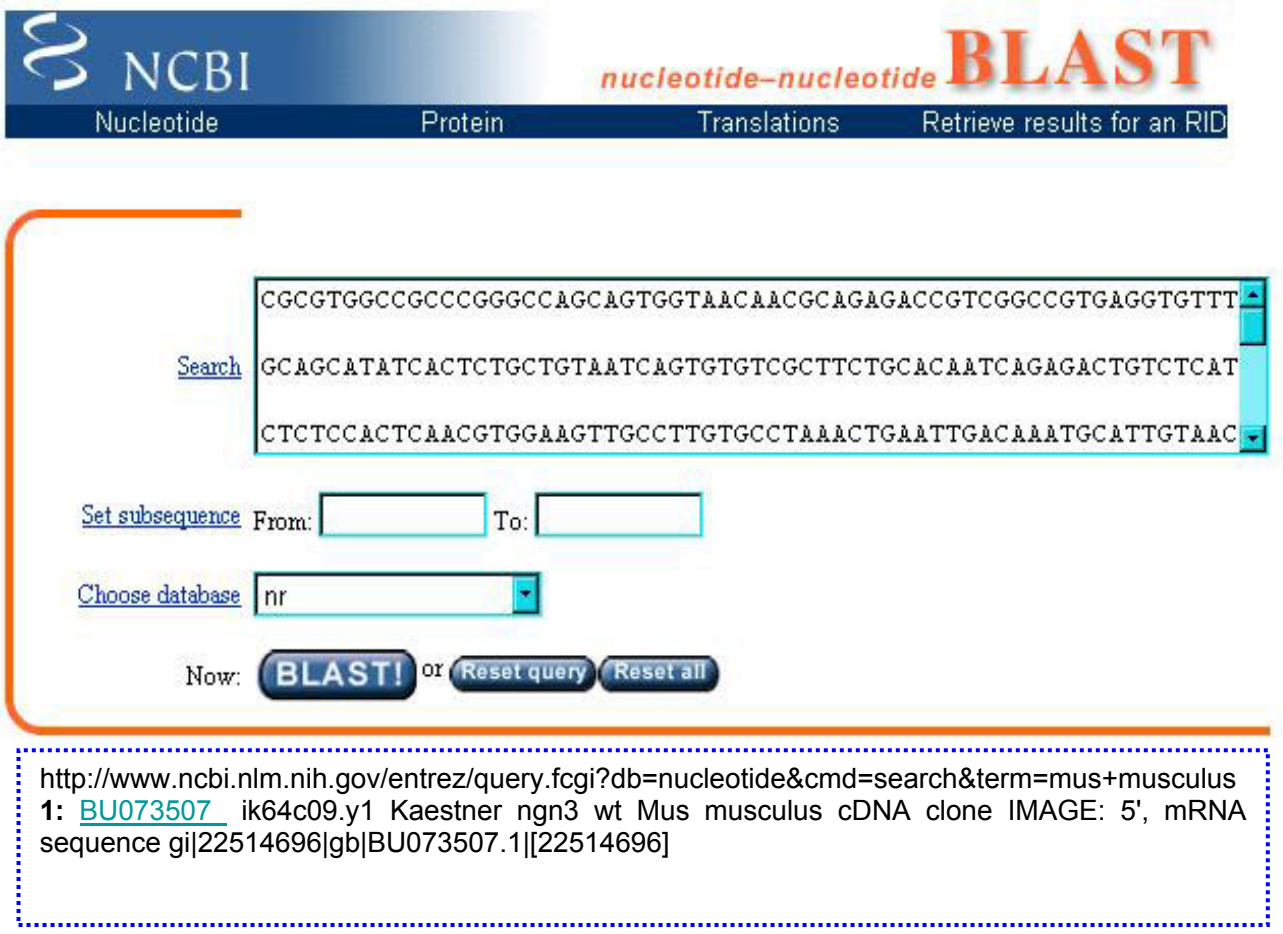

Figura 1.1-3 – Exemplo de execução do BLAST para uma seqüência de nucleotídeos.
Fonte: **www.ncbi.nlm.nih.gov**.

O problema é que qualquer resultado do BLAST pode envolver uma análise de dezenas ou centenas de *hits*[6], pois são mostradas todas as seqüências similares encontradas no banco de dados, sem opção de filtrá-las por algum valor, por exemplo: o escore em bits[7] (ou pontuação), o E-value[8], P-value[9], a qualidade PHRED[10], o tamanho da seqüência, dentre outros. Com isso, muitos pesquisadores tendem a considerar apenas as primeiras ocorrências da lista, o que pode ser um equívoco bastante grave, pois a ordem em que os registros são apresentados não garante sua importância para um domínio específico, conforme Figura 1.1-4:

Distribuiçao dos 121 hits e da seqüência de busca

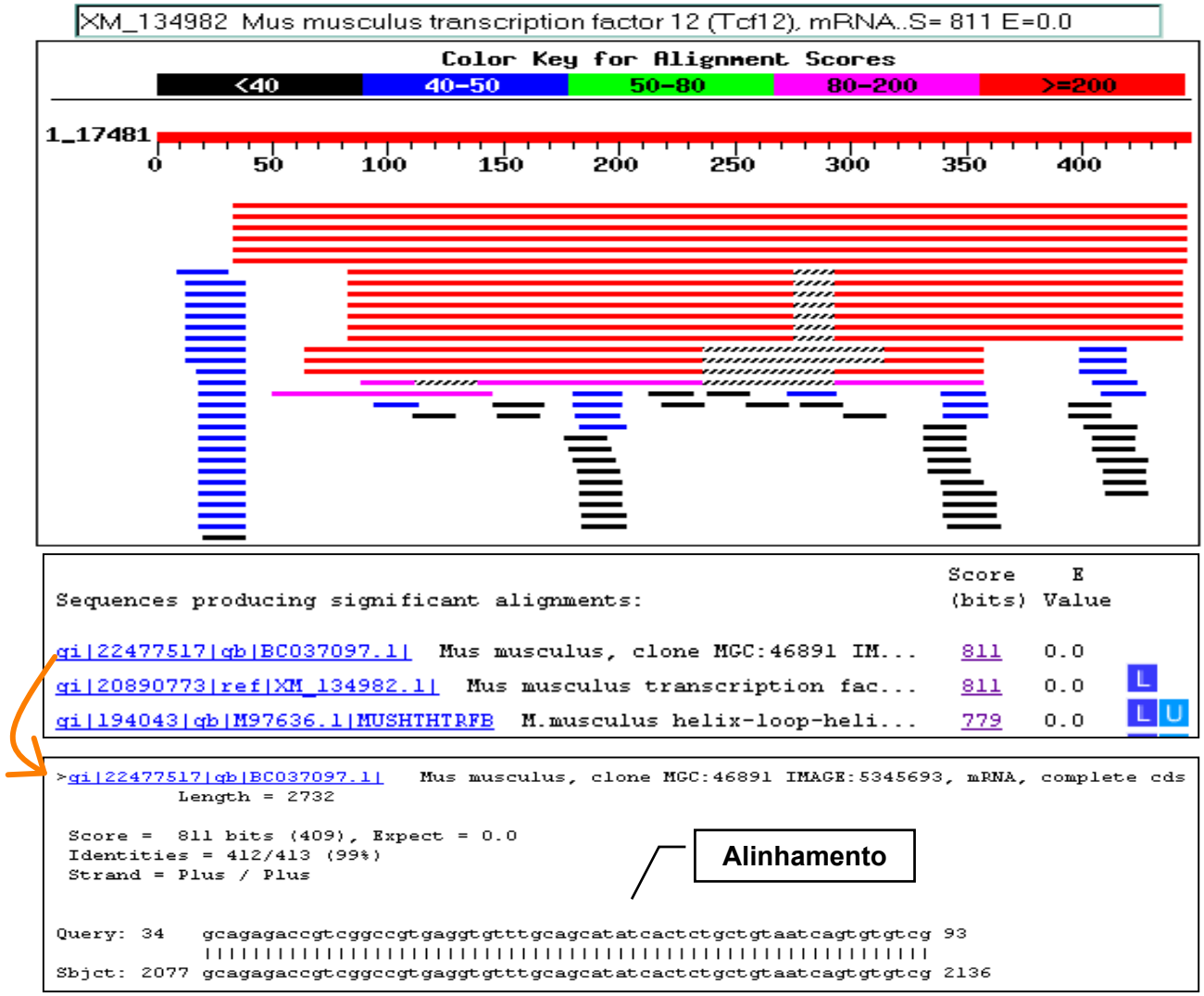

Figura 1.1-4 – Exemplo de resultado da consulta BLAST para uma seqüência de nucleotídeos.
Fonte: www.ncbi.nlm.nih.gov.

Nessa área de gerenciamento dos resultados do BLAST, quer sejam seqüências de nucleotídeos ou de aminoácidos, existe uma demanda crescente por métodos automatizados de análises básicas. Com certeza, já existem - pelo menos - três ferramentas que atendem a esse propósito: [WebBlast], [Finch-Suite] e [BioNavigator]. A primeira, pela tecnologia utilizada, já apresenta deficiências, por exemplo: informacionais, com relação às ferramentas e aos paradigmas atuais, poucos recursos de interatividade e acesso. As duas últimas, embora bastante eficazes, tornaram-se produtos comerciais e seu custo é inviável para muitas empresas, inclusive para as instituições que integram a Biofoco[11].

1.1 ORGANIZAÇÃO DO TRABALHO

Este trabalho está organizado da seguinte forma:

O capítulo 2 descreve o objeto da pesquisa, contemplando os fatores de motivação e os objetivos geral e específicos.

O capítulo 3 apresenta o referencial teórico de biologia molecular computacional , o qual embasa o entendimento sobre as análises executadas pelo aplicativo que é objeto desta Pesquisa.

O capítulo 4 descreve a metodologia utilizada para análise, projeto e implementação do aplicativo objeto desta Pesquisa. Dentre as metodologias empregadas, destacam-se: UML, RUP, JAVA, CORBA e alguns de seus serviços, e, por fim, a linguagem XML, que será utilizada para integração entre aplicativos e persistência de dados.

O capítulo 5 contempla a visão geral do projeto e o emprego das diversas metodologias citadas anteriormente.

O capítulo 6 assinala os resultados ou ganhos alcançados pela pesquisa tanto para a Bioinformática quanto para a Informática.

O capítulo 7 apresenta a conclusão desta Pesquisa, mesmo estando o Sistema BLOOM em fase inicial de implementação.

O capítulo 8 destaca os possíveis trabalhos futuros que serão legados ao informata.

O capítulo 9 lista as referências bibliográficas utilizadas nesta Pesquisa. Em seguida, são apresentados anexo e glossário.

2. OBJETO DA PESQUISA

2.1 MOTIVAÇÃO

O contínuo desencadeamento de projeto genoma por laboratórios do exterior, e recentemente com muita ênfase no Brasil, indica que a quantidade de dados biológicos disponíveis irá aumentar exponencialmente, superando os limites de se procederem análises experimentais detalhadas para todos os genes ou regiões de interesse. Sem dúvida, a bioinformática é uma das chaves para que se possa analisar esses dados **[Pappas, 2002]**.

Porém, no Brasil, a maioria das pesquisas com genoma são sustentadas por aplicativos de domínio público desenvolvidos no exterior, adaptados à realidade nacional. O Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq), percebendo a necessidade de reversão do quadro, selecionou, no ano de 2001, 28 projetos em bioinformática para financiamento, dentre eles o "Rede de pesquisa e desenvolvimento em bioinformática do Centro-Oeste - Biofoco", consórcio entre CENARGEN/EMBRAPA, Universidade de Brasília e Universidade Católica de Brasília. É coordenado por Georgios Joannis Pappas Júnior e tem a finalidade de criar uma rede de pesquisa e desenvolvimento em bioinformática que integre instituições líderes em pesquisa e ensino com foco no centro-oeste e que seja capaz de oferecer **:** apoio aos grupos e redes de pesquisa de genoma e proteoma[12]; conhecimentos; ferramentas e sistemas avançados associados a iniciativas de capacitação de técnicos e pesquisadores.

Em termos de recursos humanos, existe uma integração entre biólogos e informatas na Biofoco. Primeiramente, pesquisadores dos departamentos de computação da UCB e engenharia de redes e computação da UnB estarão participando ativamente do projeto, fornecendo tecnologia e suporte para o desenvolvimento dos sistemas de bancos de dados, softwares de análise e confecção de novos algoritmos. Um outro aspecto importante é que essa Rede possui uma grande quantidade de pesquisadores com experiência em análise de seqüências biológicas, modelagem molecular e relações entre estrutura e função de proteínas. Este ponto é de grande valia, pois solidifica a parte de mineração de dados na busca de funcionalidades para os genes encontrados. Mais

relevante ainda é a participação desses pesquisadores no processo de criação e desenvolvimento dos sistemas, dando uma contribuição decisiva no sentido de identificar os tipos de análises cabíveis, bem como auxiliando na consolidação e interpretação dos resultados **[Pappas, 2002]**.

Dentre as diversas metas da Biofoco, três delas se destacam para o propósito desta Dissertação: a) modelar e desenvolver uma arquitetura para objetos distribuídos com a utilização de padrões abertos (ex. CORBA e linguagem Java) para construção de sistemas de apoio aos projetos de genoma e proteoma; b) desenvolver um sistema para armazenamento, recuperação, anotação[13], identificação, alinhamento e outras funcionalidades para os projetos genoma e proteoma; e c) criar o "portal de bioinformática" na internet, que terá por objetivo compilar uma série de ponteiros com os serviços mais importantes na área. Na verdade, inicialmente as metas a) e b) se traduzem na implementação dos Sistemas de Anotação (em planejamento), Genoma (concluído) e BLOOM (em desenvolvimento). O Módulo de Anotação[14] também será integrado à plataforma. Todos os aplicativos serão disponibilizados aos pesquisadores por meio do portal.

Com a implementação do Sistema Genoma, foram automatizados vários procedimentos e, de modo pioneiro, viabilizou-se o estudo de genomas por diferentes equipes de pesquisadores em conjunto nas três instituições da Biofoco. Devido ao crescente interesse dos pesquisadores pelo sistema, houve demandas para que os resultados do BLAST fossem apresentados em uma interface gráfica interativa. Essas demandas e outras complementações previstas no Projeto foram agregadas, dando origem à proposta de desenvolvimento do Sistema BLOOM.

Além disso, a equipe de clientes apontou dois outros fatores de motivação inerentes ao projetos genoma realizados nas instituições envolvidas[ii]:

a) Existe um grande número de aplicativos disponíveis na Internet que executam as diversas análises propostas neste Trabalho. É possível acessá-los diretamente em alguma página da Internet ou por envio de parâmetros de linha de comando (acesso local). A questão é que o pesquisador deve empreender

[ii] Alguns dos genomas atualmente em estudo são: eucalipto, banana, café, cacau, arroz, amendoim, raízes e carrapato. Os objetivos são o melhoramento genético das espécies e combate a pragas, como é o caso do carrapato.

esforços para encontrar o aplicativo mais adequado, converter sua seqüência ao formato[15] de entrada solicitado, que pode ser diferente para cada aplicativo. Isso desestimula o uso de ferramentas variadas.

Propõe-se, então, integração do Sistema BLOOM com esses utilitários para oferecer ao pesquisador várias opções de ferramentas que possam realizar uma mesma análise, empregando um mínimo de esforço e tempo.

b) Obtido o resultado de uma certa análise, a necessidade é fazer com que esse resultado possa ser submetido a uma próxima análise com a qual possua correlação, por exemplo: resultado de alinhamento múltiplo ser submetido, em seguida, para derivação de árvore filogenética. O pesquisador enfrenta mais dificuldades: abrir uma nova janela no navegador, procurar novamente uma ferramenta adequada, converter para o formato de entrada apropriado e submenter os dados à análise. E se desejar testar várias ferramentas para uma mesma análise, o desgaste se multiplicará. Ao final, o pesquisador terá aberto várias janelas independentes, cada uma contendo resultado de uma análise. Será bastante confuso reunir as informações para um entendimento completo dos resultados ou para apresentá-las em alguma publicação científica.

Por isso, a proposta é modelar uma solução que faça com que a saída (resultado) de uma análise possa trafegar transparentemente como entrada para a próxima análise, estando todas as análises integradas em um mesmo sistema. Os resultados poderão ser analisados graficamente e obtidos também por meio de relatórios e, inicialmente, de arquivo em formato texto.

A partir desses propósitos, iniciou-se a pesquisa por ferramentas já desenvolvidas ou bibliotecas de objetos reutilizáveis que atendessem a demanda, evitando redundância de esforços. No entanto, a busca não alcançou o sucesso esperado quanto a ferramentas, por dois motivos: a) as mais completas estão disponíveis apenas comercialmente e seu custo pode se tornar inviável para a Biofoco, como é o caso do **[**Finch-Suite**]** e **[**BioNavigator**]**; e b) as de uso gratuito não atendem plenamente aos requisitos e não permitem a inserção de novos módulos, que é o caso do **[**WebBlast**]**. Porém, quanto às bibliotecas de componentes reutilizáveis, o resultado foi mais proveitoso e optou-se pelo

uso do pacote BioJava[iii], pois seus componentes de visualização de alinhamento de seqüências se mostraram bastante interativos e ricos em funcionalidades.

Quanto a outros aplicativos em estudo, embora nem todos os componentes sejam candidatos a reuso nesta primeira fase do Sistema BLOOM, o que se pretende é um ganho na avaliação das idéias empregadas, a saber:

a) BioWidgets toolkit[iv]: coleção de componentes JavaBeans para desenvolvimento de aplicações genômicas. Há componentes para visualização de resultados BLAST (BlastView[v]) e anotação de seqüências de DNA ou proteína (AnnotView[vi]). Foram desenvolvidos pela Universidade da Pensilvânia, à qual pertencem a propriedade e os direitos autorais. Pode-se conseguir licença de uso do BlastView.

Um estudo mais aprofundado desses componentes será incluído como pesquisa futura, juntamente com a utilização da plataforma J2EE, pois se observou o uso de parametrização de atributos em diversos casos, a interface gráfica é bastante interativa e também há um modelo simplificado de integração de análises. No entanto, não se possibilita ao pesquisador a escolha da ferramenta que executará a análise solicitada.

b) JalView[vii]: desenvolvido pelo Instituto Europeu de Bioinformática (EBI), é uma ferramenta escrita em Java (versão 1.1) para analisar padrões de conservação de resíduos em um alinhamento múltiplo de proteína usando o ClustalW[viii], além de ser um editor interativo de alinhamentos. Também agrupa seqüências em subfamílias (*clustering* por árvore UPGMA), calcula padrão de conservação de cada grupo, atribui cores às seqüências conforme parâmetros selecionados pelo usuário, prediz estrutura secundária de proteína utilizando o Jpred[ix], dentre outras atividades. Este editor é bastante conhecido e utilizado pelos

---

[iii] BioJava – vide http://www.biojava.org/. É um projeto de código aberto dedicado a prover ferramentas Java para processamento de dados biológicos. Isso inclui objetos para manipulação de seqüências, file parsers, interoperabilidade CORBA, DAS, acesso a ACeDB, programação dinâmica, rotinas estatísticas simples etc.

[iv] BioWidgets – vide http://www.cbil.upenn.edu/bioWidgets/index.html .

[v] BlastView – vide http://www.cbil.upenn.edu/bioWidgets/ .

[vi] AnnotView – vide http://www.cbil.upenn.edu/bioWidgets/annotViewDemo/index.html.

[vii] JalView– vide http://www.hgmp.mrc.ac.uk/embnet.news/vol5_4/embnet/body_jalview.html ou http://www.es.embnet.org/Doc/jalview/help.html

[viii] ClustalW – vide http://www.ebi.ac.uk/clustalw/.

[ix] Jpred – vide http://www.compbio.dundee.ac.uk/~www-jpred/.

pesquisadores. Algumas ferramentas também o integram em sua plataforma, por exemplo o BioNavigator.

Foram detectadas as mesmas vantagens assinaladas para o BlastView. No entanto, este editor apresenta deficiência tecnológica quanto aos recursos da linguagem Java, que atualmente já se encontra na versão 1.4, da plataforma Java2, na qual foram corrigidos alguns problemas da versão inicial e alterou-se completamente o pacote de interface gráfica para que fossem eliminados os componentes pesados (*heavyweight*) e dependentes do sistema operacional.

c) Neomorphic Genome Software Development Kit (Neomorphic Genome SDK ou NGSDK[x]): desenvolvido por Neomorphic Software Inc. da Califórnia, é uma coleção de componentes Java (versão 1.1) para visualização de mapas lineares (físico, genético, de seqüências etc), de montagem e alinhamento de seqüências, histograma de qualidade de escores etc. Esta coleção foi utilizada no Projeto Genoma Drosophila, da Universidade de Berkeley. Há disponibilidade de download do código-fonte no link https://www.neomorphic.com/das/register.html.

Ao final do ano de 2000, a companhia Neomorphic foi vendida à Affymetrix[xi], empresa que desenvolve tecnologias para a pesquisa biomédica. Com isso, houve a descontinuidade no desenvolvimento e atualização do código aberto disponível. Apesar desse fato e da deficiência quanto aos recursos da linguagem Java, foram observadas boas características de apresentação gráfica e componentes definidos em módulos bem pequenos (alta granularidade), o que facilita reuso e manutenção.

[x] Neomorphic – vide https://www.neomorphic.com/das/ e http://www.neomorphic.com/
[xi] Affymetrix – vide http://www.affymetrix.com/.

2.2 OBJETIVOS OU PROPOSIÇÕES

Em resposta à escassez de produtos de livre acesso (*freeware*), portáveis e de código aberto (*open source*), houve a sugestão dos integrantes da Biofoco para que se desenvolvesse uma ferramenta de estratégia efetiva para manipulação de informações e análise elementar nos primeiros estágios de maturação de dados de seqüências em projetos de mapeamento genômico.

Vale lembrar que o objetivo deste trabalho não é redesenvolver um outro aplicativo do tipo BLAST, embora haja pesquisas interessantes nessa área, inclusive na UNB (Universidade de Brasília) em parceria com a UCB (Universidade Católica de Brasília). Ao contrário, utilizando a ferramenta de mineração de dados (uma das maneiras de conceituar o BLAST), pretende-se que os resultados obtidos sejam tratados de maneira mais flexível do que vêm fazendo os atuais produtos de livre acesso.

Para esse tipo de mineração de dados de seqüências, além do BLAST, também há o FASTA ou FAST (Fast Alignment - **[Pearson, 2002]**). Ambos apresentam alto grau de confiabilidade dos resultados. No entanto, a escolha recaiu sobre o primeiro devido ao seu largo uso e aceitação pela comunidade científica de bioinformática. Realmente desde a sua publicação em 1990, o BLAST é um dos mais populares programas de pesquisa de semelhanças em bases de dados. **[Rocha, 2000]** Outra vantagem é que, enquanto o FASTA procura coincidências estritas de seqüências, o BLAST procura coincidências de seqüências que se assemelham (não estritamente idênticas). A noção de semelhança é incorporada no algoritmo através da utilização de uma matriz de escores, por exemplo, a matriz BLOSUM62.

2.2.1 Objetivo geral

Desenvolver um ambiente integrado para visualização e manipulação gráfica do resultado de análises genômicas, utilizando tecnologias atuais de orientação a objetos, como Java, CORBA, XML, dentre outras. Pretende-se também que a ferramenta seja**:** parametrizada, amplamente configurável, modular, capaz de adaptar-se aos diferentes projetos genoma, de fácil integração com novos componentes, estável e que mantenha a integridade e a consistência dos dados durante o processamento de análise. Inicialmente serão consideradas apresentação e manipulação gráfica dos resultados do BLAST e de

anotação, de alinhamento múltiplo, de predição de estrutura secundária de proteína e de derivação de árvore filogenética.

2.2.2 Objetivos específicos

Conforme o levantamento de requisitos, os principais objetivos específicos do sistema são:

- **a)** Integrar com: Sistema Genoma e BD *pairwise* XML (desenvolvidos pelos integrantes da Biofoco); Módulo de Anotação (desenvolvido pelo aluno André Barreto da pós-graduação da UCB); e aplicativos externos para execução das análises.
- **b)** Integrar as diversas ferramentas agregadas à plataforma para que a saída (resultado) de uma análise possa trafegar transparentemente como entrada para a próxima análise.
- **c)** Facilitar a consulta aos resultados do BLAST a partir de uma seqüência selecionada no BD *pairwise* XML.
- **d)** Executar alinhamento múltiplo a partir dos *hits* resultantes da execução do BLAST, e permitir visualização gráfica do resultado. O pesquisador poderá escolher, dentre as diversas ferramentas apresentadas, a mais adequada para a execução da análise.
- **e)** Executar predição de estrutura secundária de proteína e derivação de árvore filogenética a partir das seqüências selecionadas para o alinhamento múltiplo, e permitir visualização gráfica do resultado. O pesquisador poderá escolher, dentre as diversas ferramentas apresentadas, a mais adequada para a execução de cada uma das análises.
- **f)** Permitir visualização mais interativa dos alinhamentos do *hits*, por exemplo: fazer com que a seqüência de busca fique ancorada no topo da tela e não deslize juntamente com as demais, visualização do alinhamento *pairwise*[16] na mesma janela sem perder a visualização completa do alinhamento dos *hits* etc.
- **g)** Sempre que possível, ajustar cores e outras características das seqüências de acordo com critérios do tipo escore, e-value, z-value, qualidade PHRED, dentre outros.

h) Sempre que possível, restringir os resultados de acordo com os parâmetros especificados pelo pesquisador. Para isso, será implementado o Serviço de Filtragem.

i) Permitir ao pesquisador consultar diretamente novos resultados armazenados no BD *pairwise* XML, pelas escolhas constantes em sua lista de preferências[17].

j) Agendar e executar automaticamente atividades relativas às seqüências de interesse armazenadas no BD *pairwise* XML, por exemplo uma nova consulta BLAST.

3. REFERENCIAL TEÓRICO DE BIOLOGIA MOLECULAR COMPUTACIONAL

3.1 NOÇÕES BÁSICAS DE BIOLOGIA MOLECULAR

Devido ao propósito desta Dissertação estar voltado especificamente para bioinformática[18], apenas alguns conceitos básicos de biologia molecular de eucariotos[19] são mencionados neste tópico, visando a subsidiar o entendimento do leitor quanto aos assuntos de biologia computacional tratados em seguida.

3.1.1 A célula

No mundo real, todos os organismos vivos consistem de células, e cada célula contém um conjunto de um ou mais *cromossomos* (*strings* de DNA) e que servem como um anteprojeto do organismo. Um cromossomo contém *genes*, que são os blocos funcionais de DNA e cada um dos quais codifica uma proteína. Grosso modo, pode-se pensar no gene como sendo a codificação de um traço genético, tal como a cor dos olhos. As diferentes possibilidades de configuração de um traço (azul, castanho etc) são chamadas *alelos*. Cada gene está localizado em um posição particular (*locus*) no cromossomo **[Mitchell, 1996]**.

Muitos organismos têm múltiplos cromossomos em cada célula. A completa coleção de material genético (todos os cromossomos juntos) é chamada de *genoma*. O termo *genótipo* se refere a um particular conjunto de genes contidos em um genoma. Dois indivíduos que têm idênticos genomas são ditos terem o mesmo genótipo. O genótipo é a base para o *fenótipo*, características tais como cor dos olhos, altura, cor da pele, tipo de cabelo e outras **[NCBI/Primer, 2002]**.

Organismos cujos cromossomos estejam ordenados em pares são chamados *diplóides*; já aqueles que possuem cromossomos simples são chamados *haplóides*. Na natureza, a maioria das espécies sexualmente reproduzíveis são diplóides, incluindo os seres humanos, que possuem 23 pares de cromossomos em cada célula somática[20]: 22 pares autossomos + 1 par de heterossomos ou cromossomos sexuais (XY no sexo masculino ou XX no sexo feminino). Durante a reprodução sexuada, ocorre a recombinação (ou *crossover*) : a) de cada um dos pais, genes dos pares de cromossomos são trocados para formar um *gameta* (um cromossomo simples); e b) esses se juntarão a

outros para criar um conjunto de cromossomos diplóides. Na reprodução sexuada haplóide, ocorre apenas a primeira etapa descrita para a reprodução diplóide. Os descendentes são os que estão sujeitos a *mutação*, na qual alguns nucleotídeos (bits elementares do DNA) são mudados em relação ao material genético herdado dos pais e, quase sempre, representam erros de cópia **[Mitchell, 1996]**.

Um outro ponto é a *aptidão (ou adaptabilidade)* de um organismo. Tipicamente podemos defini-la como sendo a probabilidade de que um organismo viverá para se reproduzir (*viabilidade*) ou como uma função do número de seus descendentes (*fertilidade*).

3.1.2 Ácidos nucléicos

I. DNA

O DNA ou ADN (*DeoxyriboNucleic Acid* ou Ácido Desoxirribonucléico) é uma dupla hélice de cadeias polinucleotídicas[21] antiparalelas[22] interconectadas pela energia cooperativa de muitas pontes de hidrogênio que se estabelecem entre as bases nitrogenadas complementares dos nucleotídeos, as púricas[23] (Adenosina e Guanina) e as pirimidínicas[24] (Citosina e Timina). Cada nucleotídeo consiste de três partes: uma base nitrogenada, uma molécula de açúcar (a desoxirribose) e o resíduo de fosfato. Cada base projeta-se para o interior da hélice a partir dos esqueletos externos de açúcar-fosfato, ligando-se à base complementar na fita oposta. São, desse modo, chamadas bases complementares, pois estão sempre ligadas duas a duas, A com T e C com G, formando pares (pares de base ou *bp-base pair*). Essa regra de pareamento é o que garante a formação de pontes de hidrogênio (duas pontes de hidrogênio entre A e T e três entre C e G).

Os nucleotídeos são geralmente abreviados pela primeira letra e escritos em seqüência linear da forma CCTATAGGCA, por exemplo. A seqüência de bases é direcional da extremidade 5´ (*upstream*[xii]) para a extremidade 3´ (*downstream*[xiii]). Segue a Figura 3.1-1:

[xii] Upstream - palavra em inglês ampla e costumeiramente utilizada no jargão de biologia.
[xiii] Downstream - palavra em inglês ampla e costumeiramente utilizada no jargão de biologia.

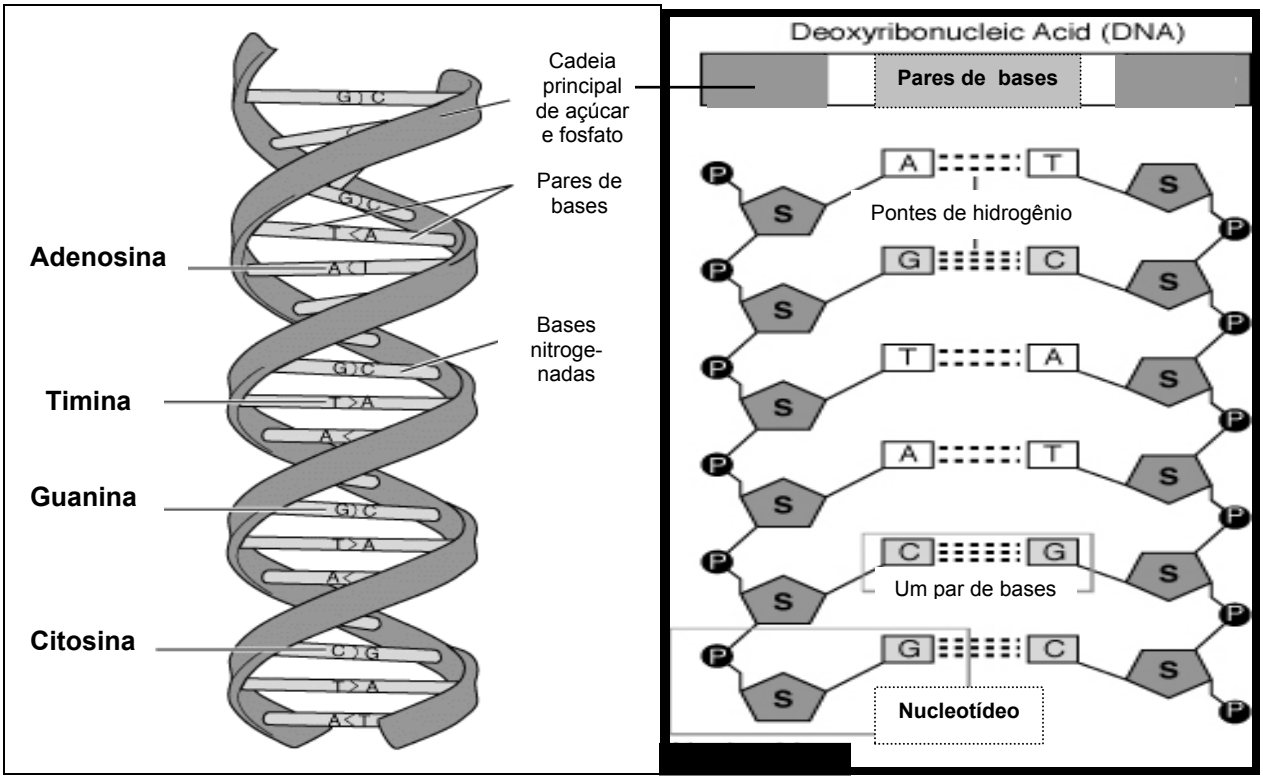

Figura 3.1-1 – A dupla hélice do DNA.
Do lado **esquerdo**, a fita dupla do DNA e suas bases nitrogenadas. Fonte: http://www.genome.gov/Pages/Hyperion//DIR/VIP/Glossary/Illustration/dna.shtml. Do lado **direito**: detalhamento da estrutura do DNA. Fonte: http://www.nhgri.nih.gov/DIR/VIP/Glossary/Illustration/gene2.html.

As moléculas de DNA costumam ser bastante longas. Em células humanas, chegam a atingir centenas de milhares de nucleotídeos. Conforme afirma **[Hunter, 1993],** a seqüência completa que descreve uma pessoa pode chegar a conter tantos caracteres quanto três anos de edições completas do New York Times (aproximadamente $3x10^9$). Para serem acomodadas no núcleo da célula, essas seqüências enovelam-se pelo auxílio das histonas[25], constituindo, por fim, o cromossomo (Figura 3.1-2).

O DNA pode assumir uma variedade de conformações ou formatos. Na maior parte das circunstâncias, ele forma a clássica hélice dupla, chamada B-DNA (Figura 3.1-2); já em outras circunstâncias, entretanto, ele pode tornar-se muito enovelado ou mesmo inverter a direção das fitas, o Z-DNA. Essas formas alternativas fazem com que um gene se torne ativo ou não, evidenciando que a geometria da forma B-DNA também pode ser explorada pelos mecanismos da célula. Isso pode ser uma importante lição para o cientistas: há mais informação a ser inferida de uma célula do que aquela disponível nos bancos de dados **[Hunter, 1993]**.

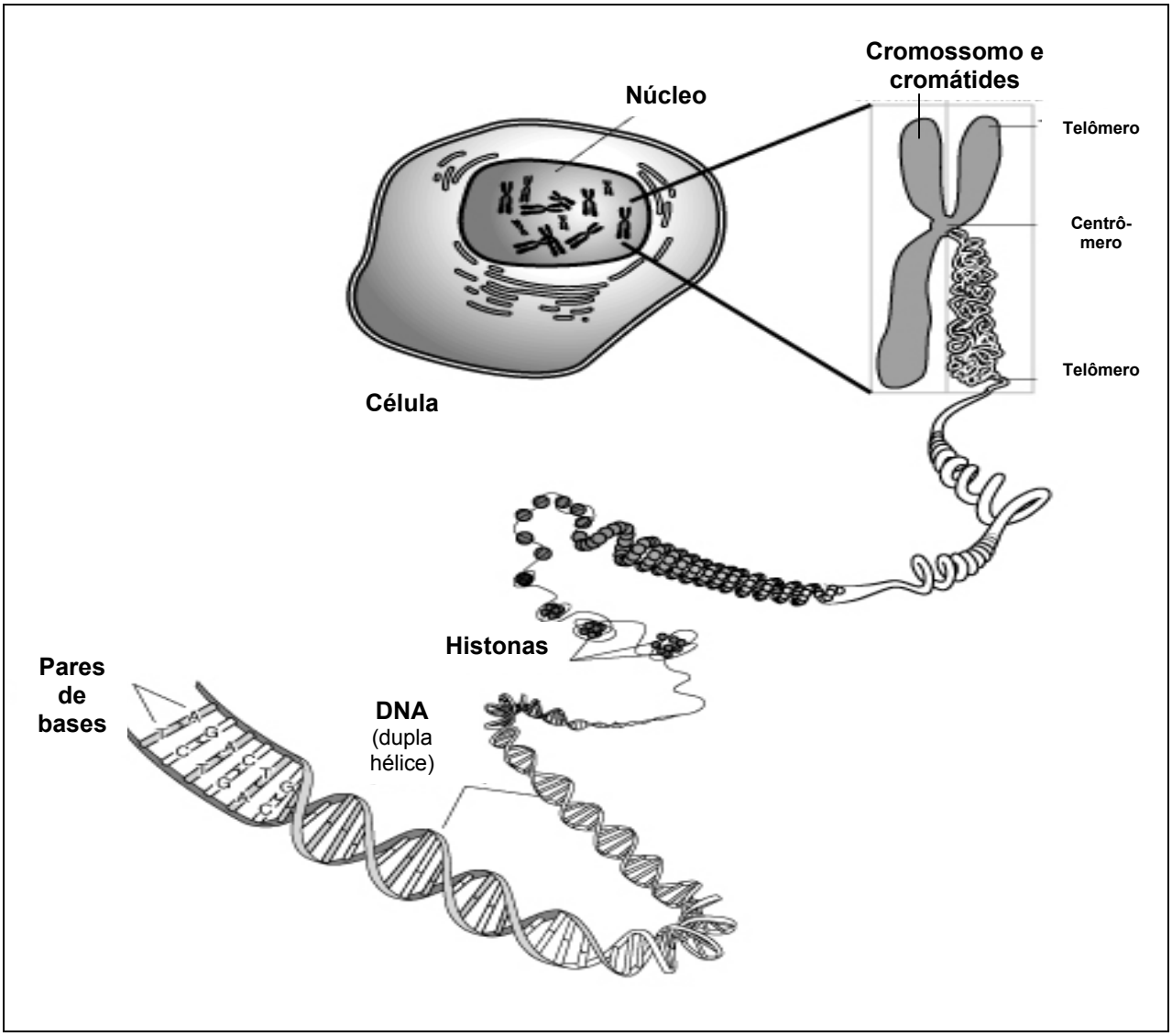

Figura 3.1-2 – A estrutura do cromossomo.
São mostrados: a célula, o núcleo e o cromossomo, que inclui as cromátides, o telômero, o centrômero, as histonas, a dupla hélice de DNA e os pares de bases. Fonte: http://www.nhgri.nih.gov/DIR/VIP/Glossary/Illustration/chromosome.html.

Mais especificamente, tamanho, composição e arquitetura dessas longas fitas variam enormemente entre as espécies. **[Pereira, 2001]** Por exemplo, as bactérias possuem cerca de 2 a 5 milhões (Mega) de bp (Mb) organizados normalmente em um cromossomo circular, bem diferente dos eucariotos, que organizam o seu DNA em alguns cromossomos lineares; as leveduras e fungos, ditos eucariotos inferiores, possuem de 10 a 40 Mb; os vermes, por volta de 100 Mb; nos insetos, esse conteúdo gira em torno de 300 Mb; e o homem possui cerca de 3 bilhões de bp (ou 3 Gb). As variações de conteúdo para os diferentes grupos é enorme e a quantidade de DNA de um organismo não necessariamente reflete a sua complexidade. Nas plantas, por exemplo, a ervilha e o

milho possuem cerca de 5 Gb; e o trigo, 17 Gb. Ou seja, uma quantidade bem maior do que a presente em humanos.

A importância desse polipeptídeo é atribuída a dois fatores: a) primeiramente,conter a informação genética de todo ser vivo a ser transmitida de uma geração a outra; e, não menos importante, b) ser o molde para a síntese de proteína, conforme será explicado no tópico O Dogma Central revisado.

Dentro de um genoma, existem dois tipos básicos de seqüência: as codantes e as não codantes. Nas primeiras, estão os genes, que são seqüências de DNA transcritas em moléculas intermediárias chamadas de RNA mensageiro (mRNA), que, por sua vez, serão traduzidas pelos ribossomos segundo um código genético (cada trinca de nucleotídeo corresponde a um aminoácido) para formar as proteínas. Também nessa classe estão incluídos os genes que codificam tipos específicos de RNA, como o ribossômico e o transportador. O segundo tipo de seqüência é a não codante, na qual se incluem promotores[26], regiões de importância estrutural (origens de replicação, centrômeros, telômeros, satélites[27]) e o assim chamado, precipitadamente, de "DNA lixo"[28]. Esse último subtipo corresponde à seqüência para a qual ainda não se encontrou função definida, mas que é altamente provável que tenha alguma **[Pereira, 2001]**.

Curiosamente, quanto maior o conteúdo de DNA de um determinado organismo, maior o conteúdo de DNA não codante e, portanto, menor a densidade de genes. Por exemplo, na bactéria *Escherichia coli,* com 4,64 Mb, foram encontrados 4.397 genes, uma média de um gene a cada 1.050 bp. Nas leveduras, em 12 Mb foram localizados cerca de 6.600 genes, um gene a cada 1.800 bp; e, nos humanos, com 3.000 Mb, foram até agora identificados cerca de 40.000 genes, ou seja, apenas um gene a cada 75.000 bp. O conhecimento desses dados é essencial quando se busca realizar o seqüenciamento do genoma de um determinado organismo e, a partir dele, pode-se delinear a estratégia a se utilizar.

II. RNA

O RNA ou ARN (*RiboNucleic Acid* ou Ácido Ribonucléico) é uma molécula estruturalmente similar ao DNA, exceto pelas seguintes diferenças **[Meidanis e Setubal, 1997]** (Figura 3.1-3):

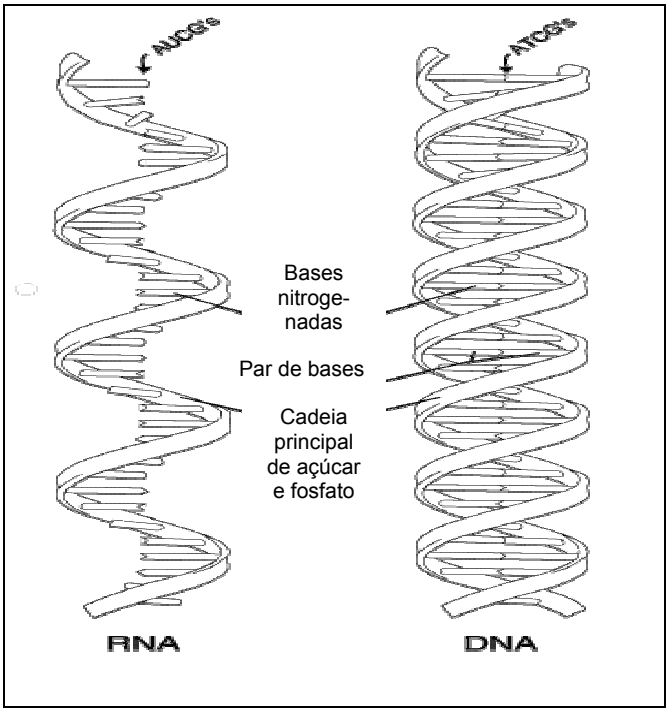

Figura 3.1-3 – RNA e DNA.
Fonte: http://www.accessexcellence.org/AB/GG/rna.html.

a) A molécula de açúcar é a ribose, ao invés da desoxirribose;

b) A timina é substituída pela uracila (U), que, do mesmo modo, liga-se à adenosina;

c) A fita, por ser simples, não forma dupla hélice. Algumas vezes, são encontradas hélices híbridas RNA-DNA; também, partes de um RNA podem se ligar a outras partes da mesma molécula por complementariedade. A estrutura tridimensional do RNA pode se mostrar até mais variada que a do DNA.

d) Existem diferentes tipos de RNA executando diferentes funções em uma célula, a saber :

- *RNA mensageiro (mRNA):* um molde para a síntese de proteína (etapa de transcrição), sendo que cada conjunto de três bases, chamado códon, especifica um certo aminoácido na seqüência formadora da proteína. A fita do mRNA é baseada na seqüência do DNA complementar.

- *RNA ribossômico (rRNA)*: junto com mais de 3 dezenas de diferentes proteínas, formam os ribossomos. Cada ribossomo é composto de duas subunidades diferentes. Na menor, há apenas um rRNA e na maior, dois. Essas subunidades estão separadas no citosol[29] e só se unem para a síntese protéica.
- *RNA transportador (tRNA)*: pequenas moléculas de RNA que agem como moléculas adaptadoras durante o processo de síntese de proteína (etapa tradução). Cada tRNA contém: em uma das extremidades, um segmento específico de três bases chamado anticódon, o qual se liga ao seu complementar códon no mRNA; e, na outra extremidade, um sítio ligante para um aminoácido específico.

A importância do RNA está diretamente ligada à ocorrência da síntese de proteína, sendo o intermediário entre o DNA e essa.

3.1.3 Proteínas

I. Aminoácidos

Constituem os principais blocos formadores das proteínas e são incorporados a essas na etapa de tradução do processo de síntese. Cada aminoácido compartilha uma estrutura básica (Figura 3.1-4), consistindo de um carbono central alfa (Cα), um grupo amino (NH_3) em uma extremidade, um grupo carboxil (COOH) noutra extremidade e um grupo R (radical), que diferencia e determina as propriedades físicas e químicas da molécula **[Meidanis e Setubal, 1997]**.

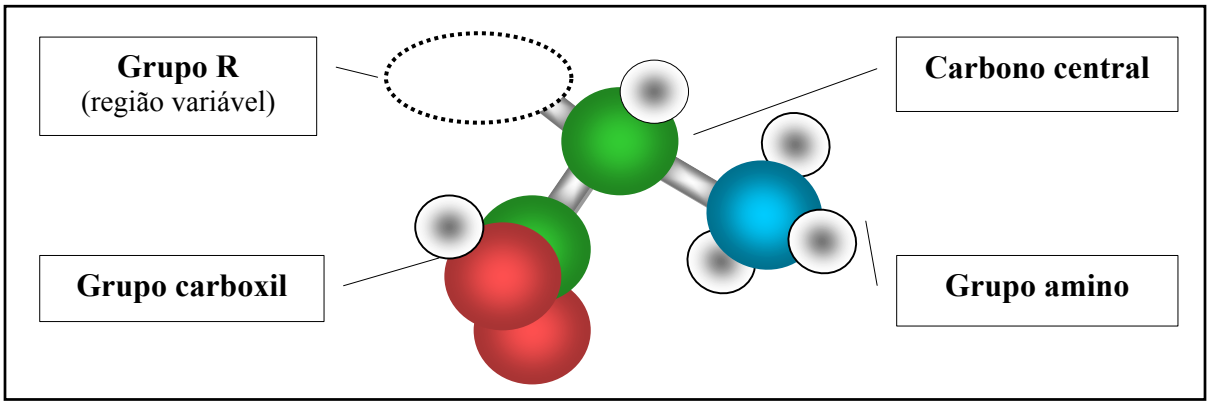

Figura 3.1-4 – Estrutura química do aminoácido.
Os carbonos estão na cor verde; os oxigênios, em vermelho; o nitrogênio, em azul; e os hidrogênios, na cor branca. As cores seguem o padrão IUPAC. Baseado em [**Hunter, 1993**].

A junção dos aminoácidos para formar a seqüência protéica é denominada ligação peptídica. Para que essa ponte seja formada, o aminoácido sofre transformações (perde dois átomos de hidrogênio e um de oxigênio – uma molécula de água), e a parte restante, que é integrada ao polipeptídeo, recebe o nome de *resíduo*. E é esse conjunto de resíduos que determina o nível primário da estrutura de cada proteína.

Na natureza, atualmente são conhecidos 20 (vinte) aminoácidos (Tabela 3.1-1), os quais são referenciados pela denominação completa, por um código de três letras ou, mais comumente, por apenas uma letra, como segue:

#	Código 1 letra	Código 3 letras	Descrição completa	Hidrofílico(□) / Hidrofóbico(◆)
1	**A**	Ala	Alanina	◆
2	**C**	Cys	Cisteína	□
3	**D**	Asp	Ácido aspártico	□
4	**E**	Glu	Ácido glutâmico	□
5	**F**	Phe	Fenilalanina	◆
6	**G**	Gly	Glicina	◆
7	**H**	His	Histidina	□
8	**I**	Ile	Isoleucina	◆
9	**K**	Lys	Lisina	□
10	**L**	Leu	Leucina	◆
11	**M**	Met	Metionina	◆
12	**N**	Asn	Asparagina	□
13	**P**	Pro	Prolina	◆
14	**Q**	Gln	Glutamina	□
15	**R**	Arg	Arginina	□
16	**S**	Ser	Serina	□
17	**T**	Thr	Treonina	□
18	**V**	Val	Valina	◆
19	**W**	Trp	Triptofano	◆
20	**Y**	Tyr	Tirosina	□

Tabela 3.1-1 – Os vinte aminoácidos comumente encontrados nas proteínas.
Baseado em http://www.library.csi.cuny.edu/~davis/Bioinfo_326/lectures/AA_Proteins/AminoAcids.htm.

Embora cada um dos diferentes aminoácidos possua características únicas, eles podem ser classificados em categorias baseadas em suas propriedades químicas, ditadas pelo Grupo R, que é a região variável. A classificação mais geral os subdivide em dois grupos : os *hidrofílicos ou polares*, que costumam ser encontrados na superfície da proteína, por exemplo glutamina; e os *hidrofóbicos ou não-polares*, encontrados no interior da proteína, por exemplo leucina e isoleucina, conforme Tabela 3.1-1 acima. Encontrar um grupo polar internamente ou um hidrofóbico exposto na superfície é uma indicação certa de que o grupo está envolvido em alguma função crítica da molécula, pois somente isso justificaria o alto dispêndio de energia para evolutivamente mantê-los assim **[Bioinfo_326, 2002]**.

Estudos de proteínas sugerem que aminoácidos com propriedades similares - citadas no parágrafo anterior - podem, mais freqüentemente, ser substituídos experimental ou evolutivamente em um certo resíduo sem alterar de forma dramática a função ou mesmo a estrutura da proteína, o que é chamado de *substituição conservativa*. Por isso, algoritmos usados para comparar proteínas sempre usam matrizes que levam em conta essa característica. As matrizes mais conhecidas são PAM 250 e BLOSUM 62, a serem estudadas mais adiante.

Resumidamente, o código genético pode assim ser mapeado:

- 3 (três) nucleotídeos especificam 1 (um) códon;
- 1 (um) códon especifica 1(um) aminoácido;
- 4 (quatro) nucleotídeos, tomados 3 (três) a 3 (três), especificam 64 (sessenta e quatro) possíveis códons;
- 64 (sessenta e quatro) códons especificam 20 (vinte) aminoácidos, 3 (três) códons de parada (STOP) e 1 (um) códon de começo (START), o qual coincide com o aminoácido metionina.

Segue a Tabela 3.1-2:

PRIMEIRA POSIÇÃO	SEGUNDA POSIÇÃO									TERCEIRA POSIÇÃO
	U	U	C	C	A	A	G	G		
U	UUU	Fenilalanina	UCU	Serina	UAU	Tirosina	UGU	Cisteína	U	U
U	UUC		UCC		UAC		UGC		C	C
U	UUA	Leucina	UCA		UAA	STOP	UGA	STOP códon	A	A
U	UUG		UCG		UAG	códon	UGG	Triptofano	G	G
C	CUU	Leucina	CCU	Prolina	CAU	Histidina	CGU	Arginina	U	U
C	CUC		CCC		CAC		CGC		C	C
C	CUA		CCA		CAA	Glutamina	CGA		A	A
C	CUG		CCG		CAG		CGG		G	G
A	AUU	Isoleucina	ACU	Treonina	AAU	Asparagina	AGU	Serina	U	U
A	AUC		ACC		AAC		AGC		C	C
A	AUA		ACA		AAA	Lisina	AGA	Arginina	A	A
A	AUG	Metionina e START códon	ACG		AAG		AGG		G	G
G	GUU	Valina	GCU	Alanina	GAU	Ácido aspártico	GGU	Glicina	U	U
G	GUC		GCC		GAC		GGC		C	C
G	GUA		GCA		GAA	Ácido glutâmico	GGA		A	A
G	GUG		GCG		GAG		GGG		G	G

Tabela 3.1-2 – O código genético mapeando códons para aminoácidos.
Baseado em **http://gened.emc.maricopa.edu/bio/bio181/BIOBK/BioBookPROTSYn.html**.

II. Proteínas: definição, funções e estrutura

São polímeros lineares ou unidimensionais, de tamanho variado, compostos por unidades mais simples, chamadas aminoácidos (unidos por ligações peptídicas na etapa de Tradução da síntese de proteína). Outros termos também usados para designar esses polímeros são: peptídeos (seqüências de até 20 aminoácidos) e polipeptídeos (seqüências com mais de 20 aminoácidos, podendo chegar até 5000). Tipicamente as proteínas costumam conter em média 300 resíduos de aminoácidos **[Meidanis e Setubal, 1997]**.

Como preconiza um adágio da Biologia Molecular, é a seqüência de resíduos de aminoácidos que determina a estrutura e esta, por sua vez, determina a função (atividade biológica) da proteína, ou melhor : as cadeias, inicialmente lineares, enovelam-se para gerar a conformação característica de cada proteína. Esse formato, juntamente com as

propriedades químicas de cada um dos resíduos de aminoácios constituintes, é importante para a função da cadeia resultante.

Subdivididas em proteínas globulares e fibrosas[30], podem desempenhar, dentre outras, as seguintes funções de **[Bioinfo 326, 2002]**:

a) **Catálise enzimática** : por meio de proteínas chamadas *enzimas*, as reações químicas que ocorrem nas células são aceleradas. Muitas dessas reações, se não acompanhadas por enzimas, poderiam levar grande tempo para finalizar ou mesmo nem acontecer.
b) **Transporte e armazenamento**: as proteínas ligam-se a outras moléculas para realizar essas atividades, por exemplo: a mioglobina liga-se ao oxigênio nas células dos músculos esqueléticos e cardíacos; a hemoglobina transporta O_2 e CO_2 nas células sangüíneas; e a ferritina faz a mediação de ferro no fígado.
c) **Construção de tecidos**: é realizada pelas chamadas proteínas estruturais que incluem a actina e a tubulina.
d) **Suporte e movimentos**: por exemplo, suporte ao fortalecimento da pele e ossos pelo colágeno (proteína fibrosa); contração dos músculos; separação dos cromossomos na mitose e na migração de células.
e) **Manutenção e expressão da informação gênica:** em todas as fases da síntese de uma proteína, muitas outras proteínas estão presentes para auxiliar o processo; regulação gênica, fatores de transcrição, dentre outros.
f) **Geração e manutenção dos impulsos nervosos:** por exemplo, as proteínas envolvidas agem como receptoras ou atuam nas sinapses[31] dos neurônios.
g) **Controle do crescimento e diferenciação da célula:** agem como fatores do crescimento; hormônio estimulante da tireóide; insulina etc.
h) **Defesa do organismo:** atuam no sistema imunológico como anticorpos contra infecções virais e bacteriológicas (imunoglobulina).

Quanto à estrutura, pode-se dizer que os quatro níveis reconhecidos são construídos a partir de associações de domínios e polipeptídeos (Figura 3.1-5), nesta ordem: nível primário, secundário, terciário (motivos[32]/domínios[33]), grupos de motivos/domínios juntos, polipeptídeos e quaternário, como se comprova a seguir **[Bioinfo 326, 2002]**:

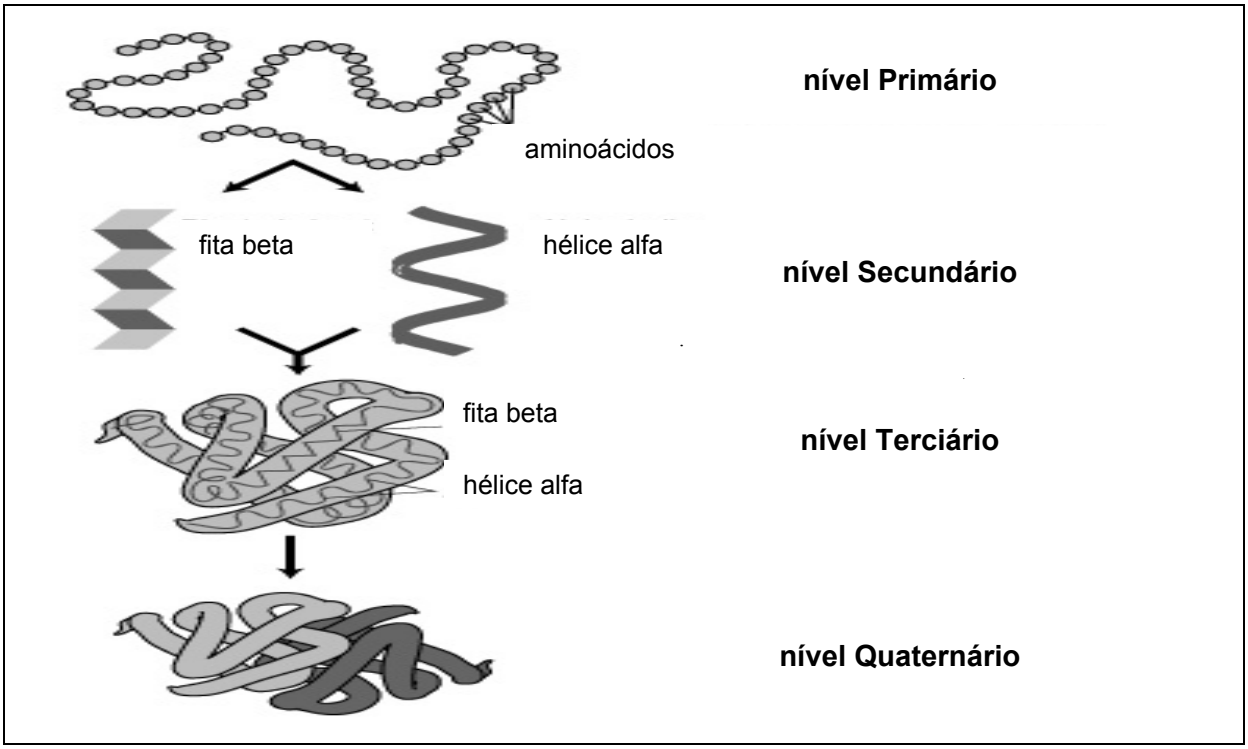

Figura 3.1-5 – Os 4 níveis estruturais de uma proteína.
Fonte: http://www.library.csi.cuny.edu/~davis/Bioinfo_326/lectures/AA_Proteins/AminoAcids.htm.

a) Primário : é a seqüência de uma corrente de aminoácidos. Pode-se distinguir a ordem seqüencial e a composição relativa (%) de cada aminoácido, conforme resultado da etapa de tradução da síntese de proteína. A convenção para se escrever a seqüência é da esquerda (aminoácido com um grupo amino alfa livre à esquerda – *amino-terminus*) para a direita (aminoácido com um grupo carboxil livre à direita – *carboxy-terminus*). Por exemplo, os 30 primeiros aminoácidos da hexoquinase[xiv] (hexoquinase de levedura da espécie *Saccharomyces cerevisiae*): "A A S X D X S L V E V H X X V F I V P P X I L Q A V V S I A ...".

b) Secundário: resulta da ligação dos resíduos de aminoácidos por pontes de hidrogênio, dando origem aos tipos : hélice α (parte externa da proteína), fita β (parte interna da proteína), segmentos circulares (em volta) e hélice tripla de colágeno.

c) Terciário: enovelamento tridimensional da estrutura e ocorre quando certas atrações estão presentes entre hélice α e fita β. São mostrados outros exemplos nas Figuras 3.1-6 e 3.1-7 a seguir.

[xiv] Mais detalhes em Brookhaven Protein Data Bank 3D browser ou SCOP (Structural Classification of Proteins), número PDB de referência 1HKG.

d) Quaternário: associação de subunidades de polipeptídeos em uma configuração geométrica definida. Se essa associação é composta de apenas uma subunidade, ela é chamada de monômero. Caso sejam múltiplas subunidades, constituirá, então, um oligômero, podendo ser de composição igual (homo-oligômero) ou diferenciada (hetero-oligômero). Podem formar estruturas complexas com mais de uma unidade, que são os dímeros (2), trímeros(3), tetrâmeros(4) e até mesmo hexâmeros(6), por exemplo rubisco, ATPase e hemoglobina. São mostrados outros exemplos nas Figuras 3.1-6 e 3.1-7 a seguir:

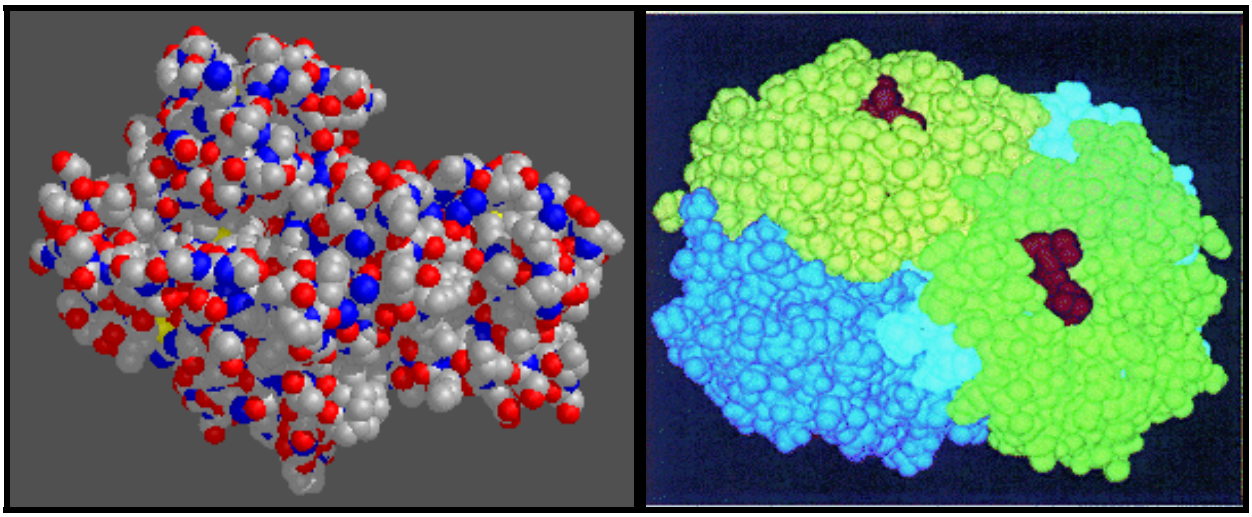

Figura 3.1-6 – Lado esquerdo: estrutura **terciária** da hexoquinase**. Lado direito:** estrutura **quaternária** da hemoglobina (com quatro polipeptídeos, sendo dois alfa-globina e dois beta-globina).
As partes em vermelho são grupos hemo (complexos de ferro ligados à proteína para transportar oxigênio).
Fonte: **http://esg-www.mit.edu:8001/esgbio/lm/proteins/structure/structure.html**.

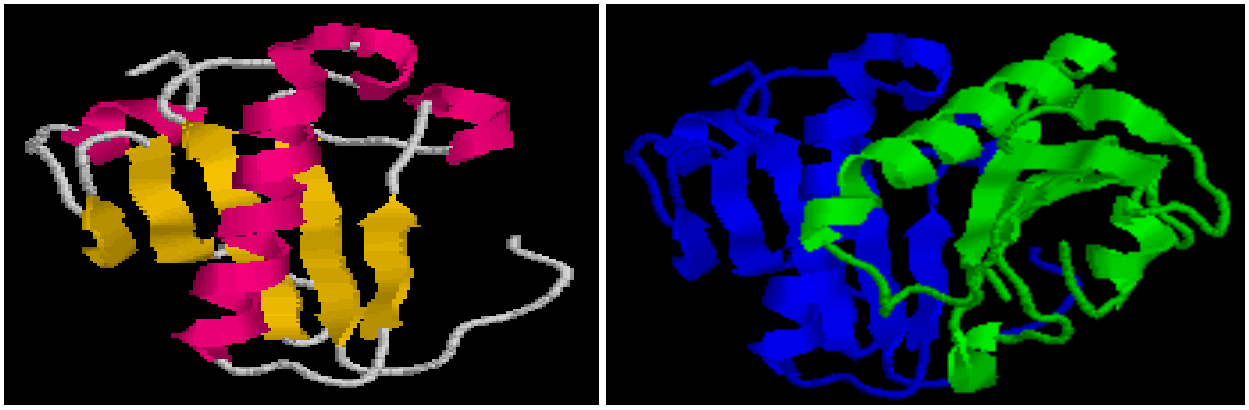

Figura 3.1-7 – Lado esquerdo: cadeia B da Protein Kinase C Interacting Protein e corresponde à estrutura em azul no outro lado. **Lado direito:** estrutura quaternária da mesma proteína.
Hélices são visualizadas como fitas e fios estendidos de betasheets por setas largas. (As figuras foram obtidas usando **rasmol** e o arquivo PDB correspondente PDB-ID 1AV5, armazenado em **PDB, the Brookhaven Protein Data Bank**).

3.1.4 O Dogma Central revisado

Fundado no início dos anos 30 pelo físico John Desmond Bernal, o laboratório de cristalografia do Instituto Cavendish (Cambridge, Inglaterra) reunia um grupo de cientistas interessados em trabalhar com a estrutura cristalográfica de moléculas. Contavam com apoio do diretor do Instituto, Sir Lawrence Bragg, prêmio Nobel de física com apenas 25 anos e cientista pioneiro na admissão da importância das técnicas de cristalografia. A partir de 1947, o laboratório tornou-se uma unidade dedicada ao estudo da estrutura molecular dos sistemas biológicos **[Bonato e Bonato,2001]**.

Por isso, em 1949, o laboratório recebeu um novo integrante: o físico Francis Crick, para desenvolver seu trabalho de doutorado sobre a estrutura da hemoglobina. No entanto, o interesse de Crick estava totalmente voltado para a molécula de DNA. Com a chegada do biológo James Watson em 1951, uma parceria foi logo estabelecida, pois ambos partilhavam a mesma inquietação de que seria a molécula de DNA, e não a de proteína, que determinaria as especificidades genéticas.

Outros cientistas mostravam-se também interessados em decifrar a estrutura do DNA, dentre eles Linus Pauling (California Institute of Technology), Rosalind Franklin e Maurice Wilkens (King's College em Londres). A partir das pesquisas desses cientistas, Watson e Crick juntaram todos os dados disponíveis em uma tentativa de desenvolver um modelo de estrutura de DNA: Franklin e Wilkens levaram fotomicrografias de difração radiográfica de DNA cristalino; sabia-se que o DNA era uma molécula longa e helicoidalmente enrolada, como determinou o trabalho de Linus Pauling; e havia também os dados de Chargaff, de apresentação do DNA como o material genético de células eucariontes **[UFRJ, 2002]**.

Então, em 1953, Watson e Crick propuseram que a molécula de DNA era constituída de uma dupla hélice de cadeias polinucleotídicas antiparalelas interconectadas pela energia cooperativa de muitas pontes de hidrogênio que se estabeleciam entre bases complementares, púricas (A e G) e pirimidínicas (C e T), dos nucleotídeos. Nesse modelo, as bases projetavam-se para o interior da hélice a partir dos esqueletos externos de açúcar-fosfato. Na tentativa de encontrar aquele que se adequasse aos dados de difração de Raio X, à complementaridade de bases e à estrutura química de cada base, Watson e

Crick chegaram até mesmo a confeccionar modelos moleculares em arame **[Bonato e Bonato,2001].**

Dando continuidade a suas pesquisas, Francis Crick e colaboradores, em 1958, propõem o Dogma Central da Biologia Molecular, afirmando que a molécula de DNA seria o molde para se construir RNA e este, por sua vez, seria o molde para a síntese de proteínas. Mostrou-se ainda a particularidade de que a informação contida na proteína não poderia ser repassada para a construção de outra proteína nem para a construção de uma molécula de DNA ou RNA, ou seja, a informação genética deveria estar armazenada em DNA e fluir daí, num sentido único, para o nível de RNA e, por fim, para o nível de proteína. É bem verdade que os biológos não tinham esse dogma como infalível **[Bonato e Bonato,2001]**.

Na década de 70, contudo, dois grupos distintos, o de Howard Temin e o de David Baltimore, mostraram que certos vírus a RNA eram capazes de, empregando o RNA como molde, fazer uma fita simples de DNA. Essa observação foi mal recebida pela comunidade científica, em especial por J. Watson. Entretanto, após sua confirmação por muitos autores em diferentes sistemas virais, ficou definitivamente demonstrado que o caminho inverso na informação gênica (entre RNA e DNA) de fato existia na natureza. O dogma passou então a ser grafado DNA $\leftrightarrow$ RNA $\rightarrow$ Proteína, pois se admitiu a transcrição reversa **[GENTROP, 2001]**.

Não se concebe a probabilidade de que a segunda parte do dogma (RNA $\rightarrow$ Proteína) seja quebrada na natureza. Conforme **[GENTROP, 2001],** a razão disto é simples: para cada base de DNA, há uma e somente uma base de RNA que pode a ela se emparelhar, e vice versa. Isto é chamado, pelos matemáticos, de uma função bi-unívoca. No entanto, para o caso das proteínas, o mesmo não acontece, pois, para cada um dos vinte aminoácidos, existem - algumas vezes - até seis diferentes códons. É possível, então, formar 64 diferentes códons de três bases por um arranjo de 4 bases tomadas três a três. Excluídos os códons que significam sinais como início e fim de síntese protéica, mesmo assim o conjunto dos códons tem muito mais elementos que o de aminoácidos. Por isso, a determinação dos aminoácidos pelos códons não é uma função, e sim uma relação, não admitindo, portanto, inversa. A Natureza, contudo, é extraordinariamente criativa e não será surpresa se for descoberto algum organismo que seja capaz de violar a

segunda parte do dogma, pois já se conhecem DNA e RNA que possuem atividade enzimática[xv].

A seguir, o esquema e as etapas do processo de síntese de proteína (Figura 0-1), segundo **[NCBI/Primer, 2002] e [Meidanis e Setubal, 1997]** :

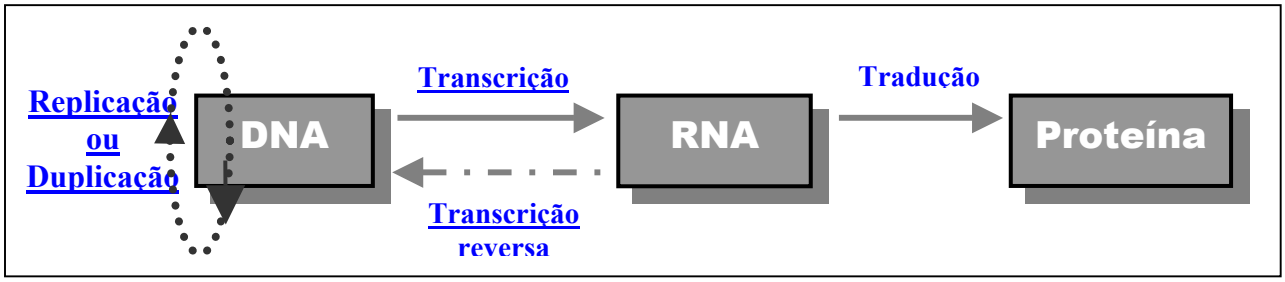

Figura 0-1 – O Dogma Central revisado da Biologia Molecular.
Baseado em **http://gened.emc.maricopa.edu/bio/bio181/BIOBK/BioBookPROTSYn.html**.

1. Replicação ou Duplicação : replicação da molécula de DNA na qual as suas duas cadeias polinucleotídicas se separam e cada uma é utilizada como molde para a polimerização de uma nova cadeia, resultando numa molécula composta da cadeia original e da recém-sintetizada. Vide Figura 0-2.

2. Transcrição: objetiva-se a produzir uma molécula de RNA mensageiro (**mRNA**). Ao longo de uma cadeia de DNA, são relacionados certos conjuntos de bases, que se arranjam de determinada maneira, com o processo de transcrição da informação contida no gene. Esses arranjos são sinais, da esquerda para a direita (sentido 5'-3'), *downstream*. Sabe-se, por exemplo, que o gene "começou" quando se encontra uma região promotora (*promoter*), que, no entendimento dos cientistas, fornece um sinal: "Logo adiante na seqüência vem um gene que deverá expressar sua proteína". Ainda *downstream*, para marcar o fim do gene, há uma região terminadora (*terminator*) **[FAPESP, 1999]**.

   O processo de transcrição ocorre do seguinte modo: uma molécula chamada *RNA polimerase* liga-se ao início de cada gene (região codante) do DNA, por indicação da seqüência promotora (*promoter*). Este processo sempre constrói moléculas de mRNA na direção 5´ (*upstream*) para

[xv] Observação sugerida pela Profa. Dra. Natália Martins.

3´(*downstream*), embora a fita-modelo seja lida na direção 3´ para 5´. A seqüência de mRNA pode ser traduzida em uma das 6 possíveis ORF´s[34], dependendo onde o processo começa. A polimerase inicia, então, uma reação na qual o DNA é utilizado como fita-modelo para a criação de uma fita complementar de RNA, chamada *transcrição primária*, pois contém éxons[35] e também íntrons. Ao final dessa seqüência, extraordinariamente 250 ou mais adenosinas (A) são adicionadas, constituindo a chamada *cauda poli-A*. Em um próximo passo (*splicing*), os íntrons de células eucarióticas são eliminados pelo *spliceossomo*, restando apenas a junção de todos os éxons para formar o *RNA maduro*, uma seqüência de três nucleotídeos chamada **códon**. O mRNA resultante é, então transportado do núcleo da célula para o citoplasma (ou citosol). Vide Figura 0-2.

3. Transcrição reversa: processo no qual uma molécula de mRNA maduro é reversivamente convertida em DNA, também chamado DNA complementar (cDNA) por não conter os íntrons. Após catalisado pela polimerase reversa, esse DNA volta ao núcleo da célula para ser integrado. Vide Figura 0-2.

4. Tradução: ocorre no ribossomo e objetiva-se a sintetizar proteína (vide Figura 0-2 abaixo). Um ou vários ribossomos ligam-se ao mRNA no códon inicial AUG (*start codon*). O processo depende da presença das moléculas de RNA transportador (tRNA), que fazem o mapeamento de códon para aminoácido. Cada tRNA possui, numa extremidade, uma conformação com alta afinidade para um códon específico e, na outra, uma conformação que se liga com facilidade ao aminoácido correspondente, sendo apenas transportado um aminoácido por vez. À medida que o mRNA passa pelo interior do ribossomo, o tRNA correspondente liga-se a esse, trazendo o aminoácido correspondente. Uma enzima apropriada junta-se ao processo para se encarregar da adição de cada aminoácido resultante à cadeia protéica, libertando-o do tRNA. Assim, resíduo por resíduo, a proteína vai sendo construída até que seja encontrado um códon de parada UAA, UGA ou UAG (stop codon) para finalizar a síntese da proteína não enovelada. A proteína, então, enovela-se em formações locais como hélices e fitas-β constituindo ligações de longo comprimento. Modificações pós-traducionais

para ajuste da função ainda podem ocorrer antes que cada nova proteína seja transportada para alguma parte da célula em que ela se faça necessária. Alguns exemplos dessas modificações são a ligação de vários grupos químicos a diferentes partes na proteína, incluindo açúcares, fosfatos, acetila, metila etc. Vide Figura 0-2 a seguir:

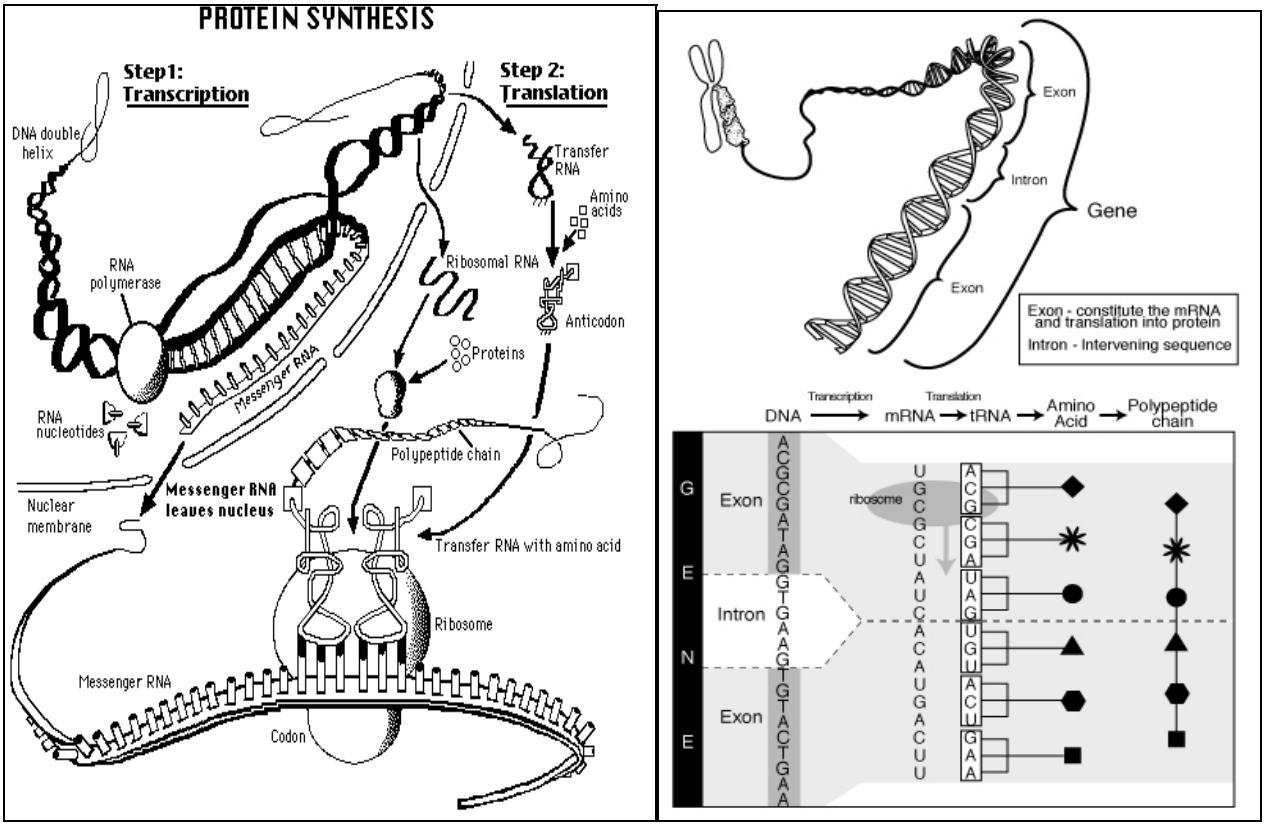

Figura 0-2 – A Síntese de Proteína.
Do lado **esquerdo**, uma visão da célula e todas organelas envolvidas na síntese de proteína. Do lado **direito**, uma visão esquemática do que ocorre no gene, uma subseqüência do DNA.
Fontes: http://gened.emc.maricopa.edu/bio/bio181/BIOBK/BioBookPROTSYn.html (lado esquerdo) e http://www.nhgri.nih.gov/DIR/VIP/Glossary/Illustration/gene2.html (lado direito).

3.2 COMPARAÇÃO DE SEQÜÊNCIAS E BUSCA EM BANCO DE DADOS

3.2.1 Tipos de comparação e a importância da busca em bancos de dados

Os problemas solucionados pela Biologia Molecular Computacional podem ser divididos em duas áreas gerais: a)comparação exata de seqüências e descoberta de padrões; e b)comparação inexata ou aproximada, que inclui a busca de similaridade por meio de alinhamento de seqüências.

Na comparação exata, **[Meidanis e Setubal, 1997]** dadas duas seqüências s (|s| = n) e t (|t| = m), o que se quer é encontrar todas as ocorrências de t em s, isto é, determinar se t é uma subsérie[36] contígua (*substring)* de s, como seria o caso de s=xaybzc e t=ybz. Diversos algoritmos têm sido empregados para atender essa demanda, sendo o mais proeminente a árvore de sufixos[37] (*suffix tree*). De maneira bastante versátil, essa estrutura tem possibilitado resolver problemas da vida real, tais como: encontrar subséries repetidas em uma seqüência de DNA; encontrar o *primer*[38] em seqüências a serem multiplicadas usando PCR[39]; e encontrar todos os palíndromos [40]de tamanho máximo em uma seqüência de DNA.

Apesar da aplicabilidade da comparação exata, é certamente a comparação inexata o caso para o qual está direcionada a grande maioria dos problemas da Biologia Molecular, dentre eles distância de edição e comparação ou alinhamento de seqüências, a chamada busca de similaridade. E será esse último ponto o foco deste Trabalho. Diferentemente da área anterior, nesta são permitidos alguns erros nas seqüências e o propósito é encontrar subseqüências[41], contíguas ou não, como por exemplo s=xaybzc e t=xyc, ou t=ayb. Ou mais especificamente, encontrar caracteres coincidentes, mesmo que esses estejam intercalados por outros divergentes ou espaços em uma seqüência combinados com caracteres na outra, o que fará com que as seqüências resultantes possuam o mesmo tamanho, uma conseqüência típica do alinhamento.

Embora esse conceito possa parecer simples, **[Gusfield, 1997]** a busca de similaridade[42] usando seqüências de DNA ou de proteínas está sendo considerada atualmente o mais poderoso método para inferir as funções biológicas de um gene (ou a proteína que ele codifica), pois uma alta similaridade da seqüência pode implicar uma

significativa similaridade estrutural ou funcional (*first fact*), embora o contrário não se aplique. Além disso, pode-se usar esse método, por exemplo, para reconstruir longas seqüências de DNA a partir de seus fragmentos (*fragment assembly*); explorar elementos informativos em DNA ou proteína; determinar o grau de mutações sofridas por uma seqüência no processo evolutivo; ou estabelecer a existência de um ancestral comum entre duas ou mais seqüências, o que é também chamado de busca de homologia [43] e será discutido mais adiante neste tutorial.

Assim, **[Meidanis e Setubal, 1997]** com o advento de tecnologias rápidas e confiáveis - como algoritmos heurísticos e computação paralela - para seqüenciamento de ácidos nucléicos e proteínas, bancos de dados centralizados foram criados para armazenar e sistematicamente organizar a larga quantidade de dados produzidos por laboratórios em todo o mundo, permitindo buscas por comparação exata ou aproximada. E, como resultado, houve uma mudança dramática no enfoque da Biologia Molecular e também da Bioquímica, pois enquanto, há vinte anos, a determinação da seqüência era geralmente um dos últimos passos na caracterização da proteína, hoje isso se reverteu. Realmente, já não se concebe a idéia de publicar uma seqüência de DNA recém-clonada sem que tenha sido feita uma busca nos principais bancos de dados.

No livro de **[Gusfield, 1997],** são apresentadas diversas histórias de sucesso, dentre elas o primeiro relato, que ocorreu ainda no ano de 1970 e foi a descoberta da correlação entre oncogenes (genes do câncer) e proteínas relacionadas ao fator de crescimento da célula; e uma outra mais recente, publicada no New York Times (03/05/1995), sobre a esclerose múltipla, na qual as imunocélulas-T utilizam as proteínas da superfície da mielina[44] para erradamente identificar as células no interior dessa como intrusas, interrompendo a transmissão normal de sinais ao longo do nervo. Por trabalhos laboratoriais e busca em bancos de dados de proteínas, concretizou-se a conjectura de que o erro acontecia porque o sistema imunológico talvez confundisse essas proteínas com outras de vírus e bactérias que teriam previamente infectado o indivíduo: usando seqüências de proteínas da superfície da mielina, realizaram-se buscas e foram encontradas aproximadamente 100 (cem) proteínas de vírus e bactérias que apresentavam alta similaridade e que também eram atacadas por células-T.

Não é sem motivo, portanto, que a indústria dos bancos de dados dessa área vem crescendo ano a ano, tanto em tamanho das bases quanto em proliferação de novos bancos. Dentre esses bancos de dados, destacam-se:

a) **Entrez** – mantido pelo Centro Nacional de Informação Biotecnológica (NCBI) nos EUA, é um sistema de busca e recuperação que integra informações de todos os bancos de dados citados abaixo (GenBank, DDBJ, EMBL, PIR, PDB e Swiss-Prot), dentre outros. Suas principais bases são: seqüências de nucleotídeos e de proteínas, estruturas macromoleculares tridimensionais, genomas completos, literatura médica (MEDLINE) etc. Todos os serviços, informações e tutoriais estão disponíveis no endereço http://www.ncbi.nlm.nih.gov/Entrez/.

b) **GenBank** (*Genetic Sequence Database* ou Banco de Dados de Seqüências Genéticas) – é um banco de dados de seqüências de DNA também mantido pelo NCBI. Faz parte da Colaboração Internacional dos Bancos de Dados de Seqüências de Nucleotídeos e pode ser acessado pelo site http://www.ncbi.nlm.nih.gov.

c) **DDBJ** (*DNA Data Bank of Japan* ou Banco de Dados Japonês de DNA) – mantido pelo Instituto Nacional de Genética (NIG) no Japão, é também um dos mais importantes banco de dados de seqüências de DNA. Faz parte da Colaboração Internacional dos Bancos de Dados de Seqüências de Nucleotídeos. Sua página principal é http://www.ddbj.nig.ac.jp.

d) **EMBL** (*European Molecular Biology Laboratory* ou Laboratório Europeu de Biologia Molecular) – é uma instituição que mantém vários repositórios, inclusive um banco de dados de DNA chamado Banco de Dados de Seqüências de Nucleotídeos mantido pelo Instituto Europeu de Bioinformática (EBI) em Cambridge, UK. Também faz parte da Colaboração Internacional dos Bancos de Dados de Seqüências de Nucleotídeos. Seus serviços estão disponíveis no site http://www.ebi.ac.uk/.

e) **PIR** (*Protein Identification Resource* ou Fonte de Indentificação de Proteína) – é um banco de dados de seqüências de proteínas cooperativamente mantido e distribuído pela Fundação Nacional de Pesquisa Biomédica (USA), pelo Instituto Martinsried de Seqüências de Proteína (Europa) e pelo Banco de Dados

Internacional de Informação de Proteína (Japão). Seus serviços estão disponíveis na página http://www.gdb.org/.

f) **PDB** (*Protein Data Bank* ou Banco de Dados de Proteína) – é um repositório de estruturas tridimensionais de proteínas. Faz parte da Pesquisa Colaborativa para o Consórcio de Bioinformática Estrutural (RCSB) mantida pela Universidade Rutgers de Nova Jersey, pelo Centro de Supercomputadores de San Diego (SDSC) e pelo Instituto Nacional de Padrões e Tecnologia (NIST). Sua página principal é http://www.rcsb.org/pdb/.

g) **SWISS-PROT** - é um repositório de seqüências de proteínas mantido sob colaboração do Departamento de Bioquímica Médica de Genebra e pela Biblioteca de Dados do EMBL. Informações e serviços estão disponíveis no site http://ca.expasy.org/sprot/sprot_details.html.

A característica comum entre esses bancos de dados é que eles funcionam como repositórios, permitindo a recuperação de qualquer seqüência pela descrição ou pelo número de acesso. Os registros costumam conter, dentre outras informações, uma descrição, a seqüência propriamente dita, o nome científicio do organismo-fonte, comentários gerais e literatura associada. No entanto, há um aspecto ainda mais importante que o armazenamento e a transferência de dados, **[Gusfield, 1997]** que é a geração de novos conhecimentos que esses bancos proporcionam pela busca baseada em similaridade. Além disso, muitos desses bancos de dados são acessíveis pela Internet e existem pacotes que possibilitam ao usuário pesquisar, ver e manipular os dados das seqüências armazenadas. Dois dos mais utilizados desses pacotes são o GCG (Genetics Computer Group) e o Standen, sendo o primeiro um serviço tarifado oferecido pela iniciativa privada e o segundo, de acesso gratuito.

Embora todos sejam basicamente repositórios de seqüências, os bancos de dados diferem bastante em suas linhas de atuação. Alguns se especializam em um tipo particular de organismo ou célula, outros se concentram em funções biológicas particulares, há os que seguem uma terminologia especializada e estilo taxonômico particulares de um subcampo da biologia, também há aqueles que tentam registrar todas as mutações e diferenças (polimorfismos) que têm sido descobertas em um dado gene ou em um conjunto de genes, e alguns diferem no modo como os dados das seqüências são

armazenados e integrados com outras informações biológicas, bem como os tipos de serviços de consulta e recuperação oferecidos.

3.2.2 Similaridade e Homologia

A semelhança entre duas seqüências (nucleotídica ou protéica) de organismos diferentes pode ser explicada pela doutrina evolucionista de que todo o material genético contemporâneo possui um ancestral comum (Charles Darwin,1859). De acordo com essa doutrina, mutações entre as famílias das atuais espécies ocorreram durante o curso da evolução, que vem desde a origem do Universo (±12 bilhões de anos), seguido pela formação do Sistema Solar (±4,6 bilhões de anos) e pelos demais acontecimentos mostrados na Figura 3.2-1:

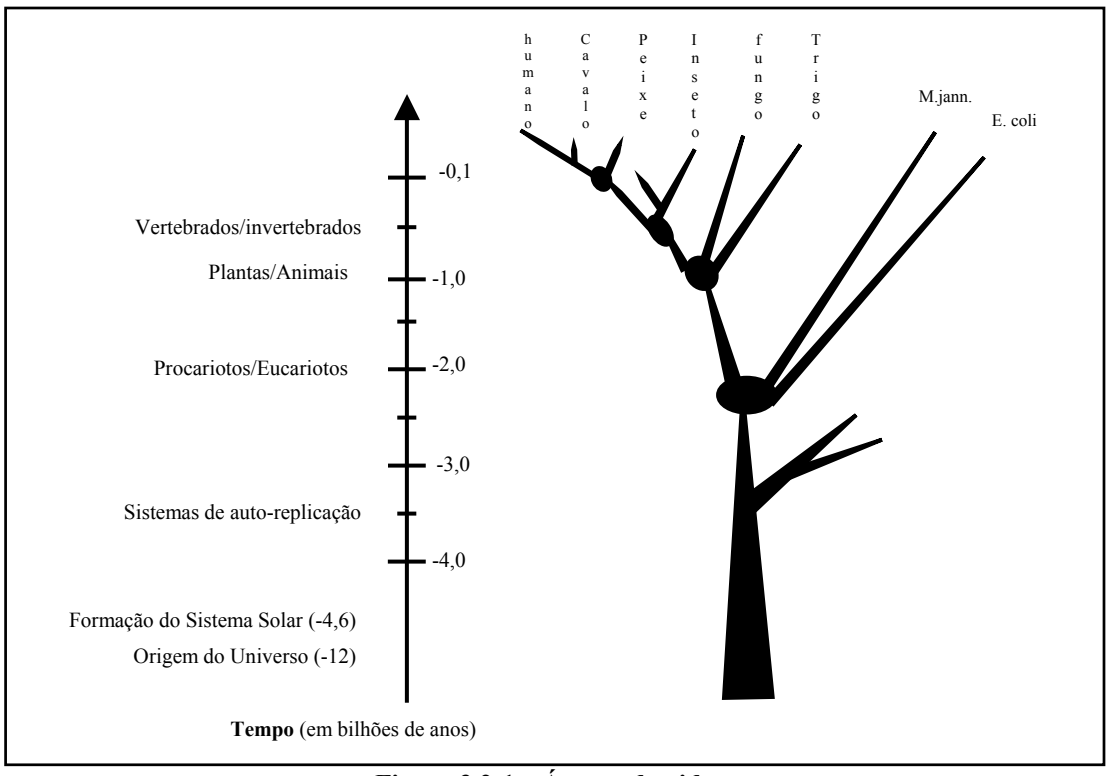

Figura 3.2-1 – Árvore da vida.
Baseada em [Pearson, 2001], p. 5.

Dessa forma, dado um alfabeto arbitrário de DNA ou de proteína, comumente ocorrem mutações locais de caracteres dentro do escopo desse alfabeto, a partir da seqüência de uma espécie ancestral, conforme exemplifica a Figura 3.2-2. Várias espécies podem resultar dessas transformações, que estão subdivididas em três categorias:

- **Inserção**: é a inclusão de um ou mais caracteres na seqüência.
- **Exclusão**: significa a retirada de um ou mais caracteres da seqüência.
- **Substituição**: é a troca de um determinado caracter da seqüência por outro do mesmo alfabeto. Quanto ao DNA, observam-se dois tipos de substituição : a) transição, que é a troca de uma base purina por outra (exemplo A→G), ou de uma pirimidina por outra (exemplo T→C); e b) transversão, que é a troca de uma base purina por uma pirimidina (exemplo A→C), e vice-versa (exemplo T→G).

Inserções e exclusões são o inverso uma da outra: dadas duas seqüências, se a inserção de um caracter (ou mais) em uma delas resulta a outra, então, equivalentemente, a exclusão desses caracteres da seqüência resultante produzirá a primeira. Devido a essa reciprocidade, essas duas operações são usualmente chamadas de ***indel*** (***in****sertion* e ***del****etion*).

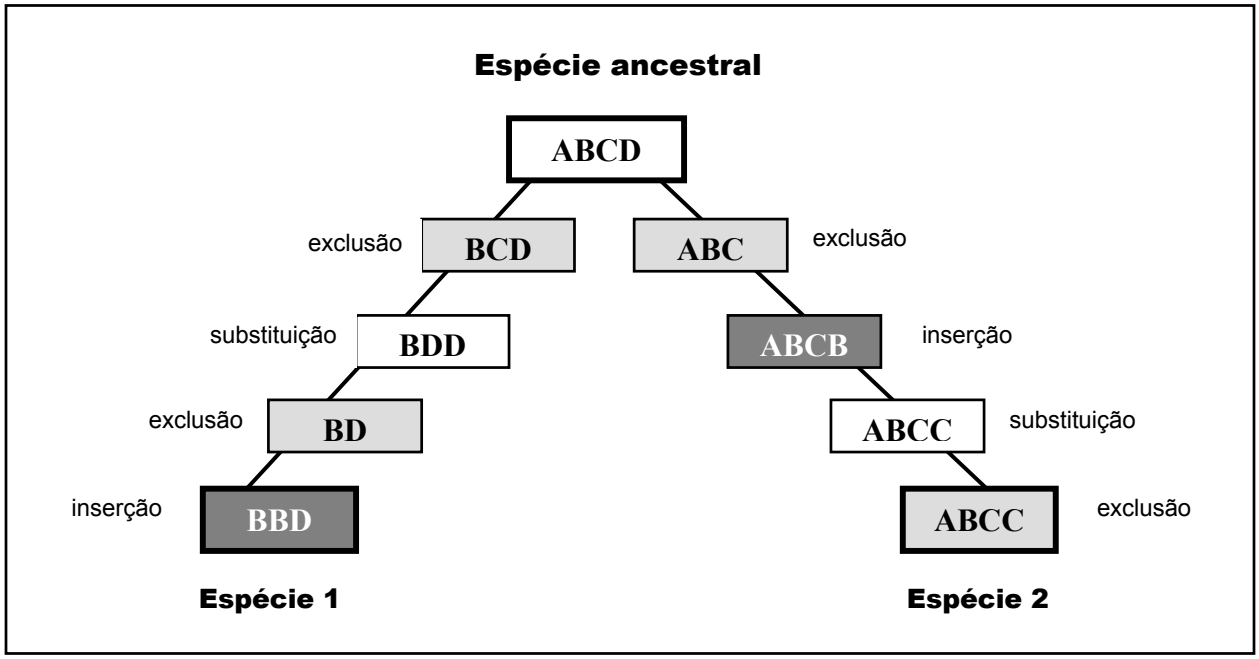

Figura 3.2-2 – Esquema simplificado de possível evolução de duas espécies a partir de um ancestral comum.

Como visto na Figura 3.2-2 acima, pode-se dizer que a maioria das seqüências são similares a outras e, se conhecemos a função de uma, podemos transferir essa informação para outras seqüências relacionadas. Mais especificamente, a similaridade ou identidade de duas seqüências pode ser subdividida em analogia e homologia. A primeira, devido à evolução convergente, faz com que haja similaridade funcional e também das seqüências sem haver um ancestral comum. Já a homologia ocorre quando essas

seqüências compartilham um ancestral comum e apresentam as mesmas funções e estrutura tridimensional (enovelamento), quer sejam seqüências similares ou não. Isso acontece porque se novas funções moleculares são necessárias, a Natureza não as projeta, mas simplesmente as produz por evolução divergente de alguns genes, a partir desse ancestral comum, conforme afirmou F. Jacob (*Evolution and tinkering. Science 196:1161-1166,1977*) : "a natureza é um produtor e não um engenheiro com uma prancheta em branco".

Embora os termos similaridade e homologia sejam usados de forma intercambiável por diversas pessoas, eles são formalmente distintos, pois o primeiro possui significado quantitativo, enquanto o segundo, qualitativo. Duas seqüências são muito similares ou pouco similares, e podem-se atribuir a elas escores, probabilidades ou valores estatísticos. No entanto, duas seqüências são apenas homólogas ou não. Além disso, seqüências similares também não são necessariamente homólogas: existem proteínas que possuem similaridade substancial e total divergência estrutural e funcional; ou o caso das seqüências análogas, citado logo acima. Contrariamente, seqüências homólogas não são necessariamente similares, como é o caso de proteínas que apresentam pouca similaridade, mas, mesmo assim, possuem funções e estruturas tridimensionais idênticas. **[Pearson, 2001]** Por isso, se as seqüências não são homólogas, nada se pode concluir a respeito de sua similaridade estrutural, mesmo que apresentem altos escores de similaridade.

Conforme mencionado no tópico anterior, um dos principais objetivos do alinhamento de seqüências, quer sejam de DNA ou de proteína, é possibilitar ao pesquisador determinar se duas seqüências apresentam similaridade suficiente, de tal modo que uma inferência de homologia seja justificada. **[Pearson, 2001]** E muitos dos algoritmos e métodos utilizados para proteínas também podem ser usados para seqüências de DNA.

Contudo, comparações de DNA geralmente são menos informativas que aquelas feitas com proteínas. Seqüências de DNA que não codificam proteínas ou RNA estruturais (por exemplo RNA ribossômico) divergem muito rapidamente e dificultam a detecção confiável de homologias para aquelas que divergiram há mais de 200 milhões de anos, o que, em contraste, não acontece mesmo com seqüências de proteína que também mudem muito rapidamente. Tipicamente comparações de seqüências de proteína

detectam outras que divergiram há 1 bilhão de anos graças à sua característica de manter regiões conservadas para também manter íntegra a função. E isso pode ser visto em um alinhamento resíduo a resíduo, no qual é freqüente que certas regiões de uma proteína, ou talvez alguns aminoácidos específicos, sejam mais altamente conservados que outros, o que pode sugerir que esses resíduos sejam cruciais para manter a função ou a estrutura da proteína. Então, se a seqüência biológica de interesse for codante, o método a escolher será sempre a comparação por proteína.

Por outro lado, podem existir posições que não sejam significantes para a função e que, todavia, mantêm-se conservadas por motivos históricos. **[Baxevanis e Ouellette, 2001]** Por isso, é preciso ter precaução quando são consideradas seqüências de espécies muito próximas, porque a similaridade pode muito mais refletir um fator histórico do que funcional, por exemplo a similaridade entre seqüências de camundongo (*mouse*) e ratazana (*rat*) homólogos, explicada apenas pelo fato de que essas séries não teriam tido tempo suficiente para divergirem de maneira evolutiva. Devido a isso, embora o alinhamento de seqüências possibilite alavancar novos conhecimentos, os autores enfatizam que todas as inferências devem ser experimentalmente testadas e não assumidas como corretas apenas com base nas análises computacionais atualmente disponíveis.

Um último ponto a ser observado quanto a homologia é sua natureza transitiva, conforme reforça **[Pearson, 2001]**. Se, por exemplo, inferiu-se homologia entre as seqüências A e B, entre B e C e, finalmente, entre C e D, então A e D também são homólogas, mesmo que não compartilhem significativa similaridade.

3.2.3 Tipos de Homologia

Seqüências homólogas podem ser divididas em dois grupos: a) seqüências ortólogas – aquelas que diferem porque são encontradas em espécies diferentes, mas conservam a mesma função; e b) seqüências parálogas – diferem por causa de um evento de duplicação de gene que faz com que apresentem função divergente **[Pearson, 2001]**.

Sem uma representação gráfica e uma explicação mais detalhada da situação, o entendimento pode tornar-se confuso para aqueles que não são especialistas na área. Por isso, um exemplo bastante ilustrativo são os membros da família de proteína globina[45] de ligação com oxigênio *(globin oxygen binding protein family)*, que são ortólogos e também parálogos (Figura 3.2-3). Assim, as globinas α do sapo, da galinha e do rato são todas ortólogas, pois elas diferem devido aos eventos de especiação que deram origem a anfíbios, aves e roedores. Da mesma forma, também sucedeu quanto às globinas β. Já a globina α do sapo e a globina β do rato, por exemplo, são parálogas devido a uma duplicação de gene que criou as subunidades α e β há aproximadamente 600 milhões de anos **[Pearson, 2001]**.

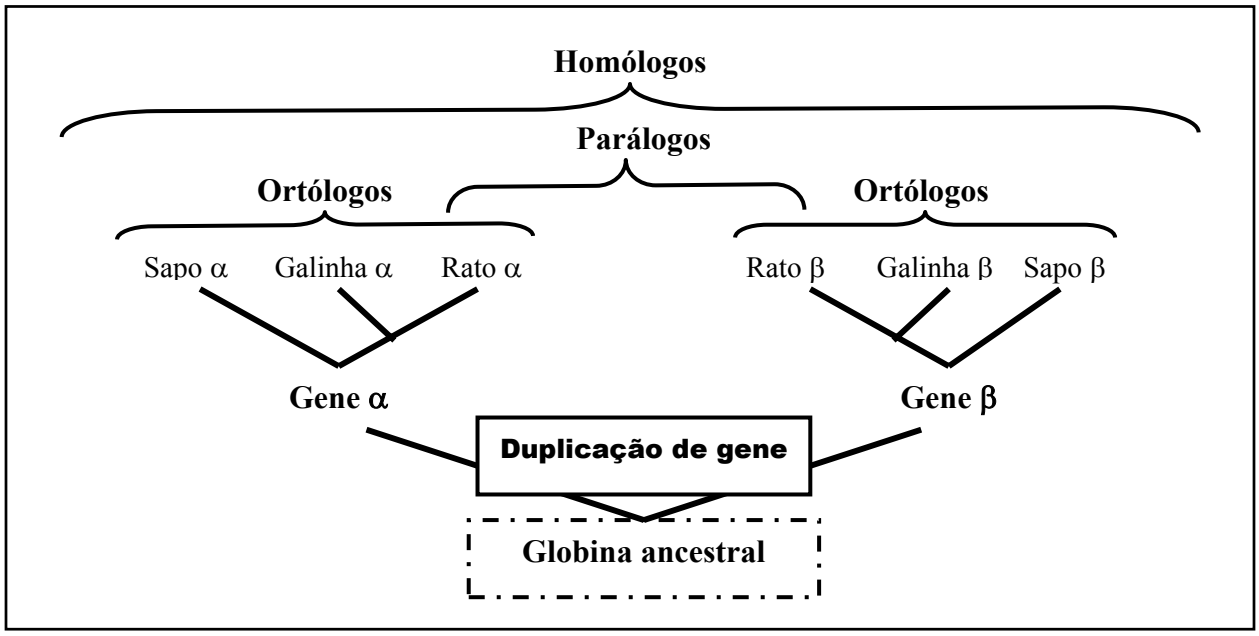

Figura 3.2-3 – Um exemplo de homologia de seqüências.
Baseado em **http://www.ncbi.nih.gov/Education/BLASTinfo/Orthology.html**.

3.3 MÉTODOS DE ALINHAMENTO

3.3.1 Medidas de distância e similaridade

Antes de se iniciar o estudo dos métodos de alinhamento e os algoritmos utilizados, é importante compreender dois conceitos que estão ligados à comparação de seqüências : distância e similaridade. Enquanto a distância se refere à medida de quanto as seqüências diferem, a similaridade é exatamente o oposto, ou seja, busca o melhor alinhamento entre duas seqüências e o escore resultante é a medida de quanto se encontrou de identidade.

Mais precisamente, a distância trata as seqüências como pontos em um espaço métrico, pois associa um valor numérico a cada par de caracteres nessas séries, o que faz com que quanto maior a distância, menor a similaridade, e vice-versa. **[Meidanis e Setubal, 1997]** Para isso, são assumidos os axiomas básicos de que o resultado não pode ser negativo; o resultado pode ser zero apenas se as seqüências são idênticas; a distância de A a B é a mesma de B a A; e a distância entre A e B é menor que a soma das distâncias de A a C e de C a B, a chamada *desigualdade triangular*.

Existem dois tipos de distância :

- **Distância Levenshtein ou de edição**: **[Gusfield, 1997]** primeiramente discutida por V. Levenshtein, é a transformação de uma seqüência em outra por meio de uma série de edições em caracteres individuais. A edição que utiliza o menor número de operações é chamada de transcrição ótima, sendo que várias dessas podem ocorrer para a mesma seqüência (transcrições co-ótimas). As operações permitidas e realizadas apenas na primeira seqüência são inserção (I) ou exclusão (E) de um caracter e substituição (S) de um determinado caracter para igualá-lo a outro da segunda seqüência. Ou equivalentemente, as operações podem ser realizadas nas duas seqüências para transformá-las em uma terceira, desde que uma inserção em uma das seqüências possa ser vista como uma exclusão na outra, e vice-versa.

  Essa medida pode ser não-valorada ou valorada. No primeiro tipo, apenas são especificadas as operações necessárias para transformar uma seqüência em outra, sem mencionar pesos para essas edições. Já no segundo tipo, para cada uma das operações citadas acima, um custo positivo é atribuído; e para as igualdades (G) de caracteres, o custo é zero. Com isso, o custo final é a soma de todas as edições ocorridas. Os dois tipos são apresentados conjuntamente nos exemplos abaixo, nos quais o custo positivo é igual a 1 (um):

  a) Para transformar “LEITOR” em “ESCRITOR” e vice-versa, uma das possibilidades seria (Tabela 3.3-1):

S	S	I	I	G	G	G	G			S	S	E	E	G	G	G	G	
L	**E**	**-**	**-**	**I**	**T**	**O**	**R**		Ou, inversamente,	**E**	**S**	**C**	**R**	**I**	**T**	**O**	**R**	
E	**S**	**C**	**R**	**I**	**T**	**O**	**R**			**L**	**E**	**-**	**-**	**I**	**T**	**O**	**R**	
+1	+1	+1	+1	+0	+0	+0	+0	= 4		+1	+1	+1	+1	+0	+0	+0	+0	= 4

Tabela 3.3-1 – Distância Levenshtein ou de edição para transformar "LEITOR" em "ESCRITOR".

b) Para transformar "AGCACACA" em "ACACACTA", seguem duas possibilidades, e a primeira é a transcrição ótima (Tabela 3.3-2):

G	E	G	G	G	G	G	I	G			G	S	I	G	G	G	S	E	G	
A	**G**	**C**	**A**	**C**	**A**	**C**	**-**	**A**		Ou	**A**	**G**	**-**	**C**	**A**	**C**	**A**	**C**	**A**	
A	**-**	**C**	**A**	**C**	**A**	**C**	**T**	**A**			**A**	**C**	**A**	**C**	**A**	**C**	**T**	**-**	**A**	
+0	+1	+0	+0	+0	+0	+0	+1	+0	= 2		+0	+1	+1	+0	+0	+0	+1	+1	+0	= 4

Tabela 3.3-2 – Distância Levenshtein ou de edição para transformar "AGCACACA" em "ACACACTA".

Matematicamente, transcrição e alinhamento são equivalentes e pode-se deduzir uma medida a partir da outra: toma-se a transcrição ótima, atribui-se um valor x para cada par de caracteres divergentes, atribui-se um valor y (menor que x) para cada ocorrência de espaços e, então,obtém-se o resultado final pela soma de todos os valores encontrados.Quanto maior o resultado final, melhor o alinhamento e maior a similaridade.

Apesar da equivalência mostrada acima, esses conceitos guardam significados distintos. A transcrição enfatiza os eventos mutacionais (caracter por caracter) que transformam uma seqüência em outra, enquanto o alinhamento apenas mostra a relação existente entre essas duas seqüências. Se o primeiro é o processo, o segundo é o produto. Além disso, diferentes modelos evolucionários formalizam-se por diferentes edições e, todavia, podem resultar um mesmo alinhamento **[Gusfield, 1997]**.

A aplicabilidade da distância de edição abrange diversas áreas, dentre elas estudos funcionais, estruturais e evolucionários de seqüências

biológicas, recuperação textual em bancos de dados e métodos de correção de escrita.

- **Distância Hamming** : é definida como o número de posições nas quais são encontrados caracteres divergentes quando duas seqüências de mesmo tamanho são comparadas. Diferentemente da medida anterior, espaços não são permitidos. Vide Tabela 3.3-3.

Seqüência **s**	A **A T**	A **G C C T**	A G **C** C **G** G **C** A
Seqüência **t**	A **T A**	A **C A T A**	A G **A** C **A** G **T** A
Distância Hamming **(s,t)**	2	4	3

Tabela 3.3-3 – Distância Hamming entre as seqüências *s* e *t*.

Devido ao fato de as medidas de distância não serem apropriadas para comparação local (envolve subseqüências e subséries), ou seja, estarem restritas apenas a comparação global (envolve toda a seqüência), um caminho alternativo para aplicações biológicas é a similaridade, pois esta baseia-se sempre em alinhamento **[Meidanis e Setubal, 1997]**.

Na verdade, vista sob outro ângulo, similaridade é o mais alto escore de qualquer alinhamento, sendo esse escore do tipo aditivo, pois se o alinhamento for particionado em blocos, o escore do alinhamento inteiro será igual à soma dos escores dos blocos. Seja Σ o alfabeto usado pelas seqüências *s* e t, e seja Σ**'** o alfabeto Σ adicionado do caracter "**-**", que denota um espaço **[Gusfield, 1997]**; então, o sistema de escore é composto de um par (p, g) cujos membros são funções *p*: Σ**'** x Σ**'** → **R** e uma penalidade *g* para os espaços. Geralmente esse valor é *g* < 0. Isso posto, pode-se atribuir um valor númerico, o escore, para cada possível alinhamento : soma-se *p(a,b)* a cada vez que *a* alinha-se a *b* em *α*, e soma-se *g* a cada vez que um caracter a é pareado a um espaço. A soma total é o escore de *α,* indicado por *escore(α).* Portanto, a similaridade entre duas seqüências *s* e *t* de acordo com esse sistema de escore é:

$$\mathrm{Sim}(s,t) = \max_{\alpha \in \mathcal{A}(s,t)} escore(\alpha),$$

onde $\mathcal{A}(s,t)$ é o conjunto de todos os possíveis alinhamentos entre *s* e *t*.

3.3.2 Principais métodos de alinhamento e alinhamento *pairwise*[xvi]

Estabelecida a diferença entre distância e similaridade, este capítulo - de agora em diante - concentrar-se-á nos métodos e algoritmos de alinhamento *pairwise*, que são a maneira pela qual a similaridade compara duas seqüências.

Um alinhamento entre *s* e *t* é uma função do tipo *p*: Σ' x Σ' → **R**, sendo que Σ é o alfabeto usado pelas seqüências *s* e *t*, e Σ' é o alfabeto Σ adicionado do caracter "**-**", que denota um espaço **[Gusfield, 1997]**. As seqüências *s'* e *t'* resultantes devem satisfazer **[Meidanis e Setubal, 1997]**:

1. |*s'*| = | *t'*|, ou seja, o tamanho da seqüência *s'* deve ser igual ao da *t'*, devido à inserção de espaços em uma ou outra, ou ambas.
2. Removendo-se todos os espaços de *s'*, obtém-se *s*.
3. Removendo-se todos os espaços de *t'*, obtém-se *t*.
4. Para cada posição *i*, apenas *s'[i]* ou *t'[i]* pode conter espaço, isto é, um espaço deve ser sempre alinhado a algum caracter e nunca a outro espaço.
5. Não obedece ao princípio da *desigualdade triangular*[46] (vide tópico Medidas de distância e similaridade), um dos axiomas básicos das medidas de distância. Isso porque distância entre s e t pode ser maior que a soma das distâncias de s a x e de x a t.
6. O escore pode ser negativo[xvii], embora raramente aconteça.
7. O escore do alinhamento entre as seqüências s e t pode não ser o mesmo entre t e s.
8. Quanto mais idênticas são as seqüências, maior será o escore do alinhamento.

Considerando-se duas seqüências *s'* e *t'* de DNA ou de proteína, os principais tipos ou métodos de alinhamento utilizados pelas aplicações biológicas são:

- **Global**: é um dos tipos mais comuns de alinhamento e recebe essa denominação porque as seqüências envolvidas são consideradas em sua globalidade. Assim, tomam-se *s'* e *t'* e coloca-se uma sobre a outra de forma

[xvi] Alinhamento *pairwise* ou alinhamento em duplas. Neste trabalho será utilizado o termo em inglês, por ser o de maior uso pelos pesquisadores da área.
[xvii] Mais detalhes em http://bioinfo.weizmann.ac.il/courses/BCG/lectures/02_*pairwise*/2.3quantitative/02distance.html.

que um caracter da primeira alinhe-se a um caracter ou espaço da segunda, e vice-versa.

Para o caso das proteínas, um alinhamento global é sempre mais significativo quando as duas seqüências são membros da mesma família e se quer deduzir suas histórias evolucionárias pelo exame de similaridade ou divergência. Por exemplo, a proteína *citocromo c*[47] tem quase o mesmo tamanho na maioria dos organismos que a produzem, e espera-se ver a relação entre citocromos de duas espécies diferentes na totalidade das duas seqüências. O mesmo também se aplica para proteínas da família *globina*, tais como *mioglobina*[48] e *hemoglobina*[49] **[Gusfield, 1997]**. E **[Baxevanis e Ouellette, 2001]** concluem que esse tipo de alinhamento é mesmo apropriado para proteínas que consistam de apenas um único domínio ou para aquelas homólogas e que não tenham divergido substancialmente.

Pode-se adotar um escore para cada alinhamento e o maior valor determinará o grau de similaridade entre as duas seqüências, sendo um valor positivo para cada par de caracteres idênticos (*match*), um valor negativo para cada par de caracteres divergentes (*mismatch*) e um último também negativo para o par que contiver um espaço (*space*), ou seja, recompensam-se as identidades e penalizam-se as divergências e espaços **[Meidanis e Setubal, 1997]**.

Para os exemplos abaixo, um escore válido seria atribuir o valor +1 para as identidades ou ***m**atchs* (**M**); -1 para as **d**ivergências ou *mismatch*s (**D**); e –2 para o par que contiver espaço ou **t**raço (**T**):

a) Para se alinhar s=LEITOR a t=ESCRITOR, duas possibilidades seriam (Tabela 3.3-4):

	D	D	T	T	M	M	M	M		Ou	T	M	T	T	T	M	M	M	M	
s'	**L**	**E**	**-**	**-**	**I**	**T**	**O**	**R**			**L**	**E**	**-**	**-**	**-**	**I**	**T**	**O**	**R**	
t'	**E**	**S**	**C**	**R**	**I**	**T**	**O**	**R**			**-**	**E**	**S**	**C**	**R**	**I**	**T**	**O**	**R**	
Escore	-1	-1	-2	-2	+1	+1	+1	+1	= **-2**		-2	+1	-2	-2	-2	+1	+1	+1	+1	= -3

Tabela 3.3-4 – Alinhamento global entre as seqüências s=LEITOR a t=ESCRITOR.

b) Para se alinhar s=CGGATTAC a t=CGGATTCA, duas possibilidades também seriam (Tabela 3.3-5):

	M	M	M	M	M	M	D	D		Ou	M	M	M	M	M	M	T	M	T	
s'	**C**	**G**	**G**	**A**	**T**	**T**	**A**	**C**			**C**	**G**	**G**	**A**	**T**	**T**	**A**	**C**	**-**	
t'	**C**	**G**	**G**	**A**	**T**	**T**	**C**	**A**			**C**	**G**	**G**	**A**	**T**	**T**	**-**	**C**	**A**	
Escore	+1	+1	+1	+1	+1	+1	-1	-1	= **+4**		+1	+1	+1	+1	+1	+1	-2	+1	-2	= +3

Tabela 3.3-5 – Alinhamento global entre as seqüências s=CGGATTAC a t=CGGATTCA.

- **Local**: é um tipo mais rebuscado de alinhamento e o propósito é encontrar e extrair um ou mais pares de regiões, um de cada uma das duas seqüências, que exibam alta similaridade: dadas duas seqüências *s* e *t*, encontrar subseqüências *α* e *β* de *s* e *t* respectivamente, cuja similaridade seja máxima para todos os pares de subseqüências **[Gusfield, 1997]**.

Apesar da importância do alinhamento global, muitas aplicações biológicas têm o alinhamento local como sendo muito mais significativo. **[Baxevanis e Ouellette, 2001]** Isso é particularmente verdadeiro nos casos em que: longas seqüências de DNA são comparadas, desde que apenas algumas seções internas estejam relacionadas; seqüência de nucleotídeos no mRNA (sem íntrons) são comparadas à sua seqüência genômica, onde cada éxon estaria em um alinhamento local distinto; dada a natureza modular das proteínas, pode-se desejar procurar por regiões conservadas e de alta similaridade (sítios ativos, motivos/*motifs,* domínios, estruturas funcionais eqüivalentes etc) separadas por outras que acumulam mutações e não interferem na funcionalidade, quer sejam em famílias muito diferentes ou em seqüências homólogas, por exemplo : a)**[Gusfield, 1997, p. 231]** nas proteínas codificadas pelos genes *homeobox*, que podem ocorrer em uma larga variedade de espécies (desde a mosca da fruta a sapos ou humanos) e regulam o desenvolvimento embrionário, os *homeodomínios* conservados – não toda a seqüência - chegam a apresentar 50% a 95% de identidade sem espaços em certas espécies; b) os diversos módulos de domínios comuns (domínio de atividade da serino-protease (K), fator parecido ao do

crescimento epidérmico (E), fribonectina etc) entre o fator de coagulação sangüínea F12 e o ativador plasminogênio do tecido humano **[Baxevanis e Ouellette, 2001, p. 190]**.

Ainda quanto às proteínas, o alinhamento local é adicionalmente importante porque, diferentemente do alinhamento global, pode detectar que caracteres particulares isolados de seqüências relacionadas podem estar mais conservados que o resto da mesma proteína, por exemplo o caso dos aminoácidos do sítio ativo de uma enzima ou aqueles do centro hidrofóbico de uma proteína globular. Um bom exemplo são as serino-proteases, onde alguns poucos e isolados caracteres identificam a família. Um outro exemplo é o motivo hélice-volta-hélice (*helix-turn-helix*), que ocorre freqüentemente nas proteínas que se ligam ao DNA na regulação da etapa de transcrição e normalmente apresentam o aminoácido glicina na décima posição, enquanto o resto da seqüência é bastante variável **[Gusfield, 1997, p. 231]**.

Neste tipo de alinhamento, também se pode adotar um escore para as subseqüências, seguindo o modelo do alinhamento global. Então, simplificadamente, para as subseqüências abaixo, um escore válido seria atribuir, por exemplo, o valor +2 para as identidades ou ***m**atchs* (**M**); -2 para as **d**ivergências ou *mismatch*s (**D**); –1 para o par que contiver espaço ou **t**raço (**T**); e **i**gnorar os caracteres que estiverem fora dos limites das subseqüências de interesse (**I**), baseado em **[Gusfield, 1997]**:

a) Alinhar as subseqüências *α*=AXABCS e *β*=AXBACS de *s*=PQRAXABCSTVQ e *t*=XYAXBACSLL, respectivamente (Tabela 3.3-6):

	I	I	I	M	M	T	M	T	M	M	I	I	I	
S'	**P**	**Q**	**R**	**A**	**X**	**A**	**B**	**-**	**C**	**S**	**T**	**V**	**Q**	
T'	**-**	**X**	**Y**	**A**	**X**	**-**	**B**	**A**	**C**	**S**	**L**	**L**	**-**	
Escore	0	0	0	+2	+2	-1	+2	-1	+2	+2	0	0	0	= **+8**

Tabela 3.3-6 – Alinhamento local entre as seqüências *s*=PQRAXABCSTVQ e *t*=XYAXBACSLL.

- **Semiglobal**: [Meidanis e Setubal, 1997] adota a mesma estratégia do alinhamento global, apenas com a exceção de que o escore pode ignorar alguns dos espaços antes do primeiro e depois do último caracter da seqüência, fazendo com que o resultado não seja prejudicado pela presença desses espaços que apenas visam a igualar o tamanho das duas seqüências, como se pode ver a seguir:

  a) Alinhar as seqüências *s*=CAGCACTTGGATTCTCGG e *t*=CAGCGTGG (Tabelas 3.3-7 e 3.3-8):

	I	I	I	M	M	T	M	D	M	M	M	I	I	I	I	I	I	I	I	
s'	**C**	**A**	**G**	**C**	**A**	**-**	**C**	**T**	**T**	**G**	**G**	**A**	**T**	**T**	**C**	**T**	**C**	**G**	**G**	
t'	**-**	**-**	**-**	**C**	**A**	**G**	**C**	**G**	**T**	**G**	**G**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	
Escore	0	0	0	+1	+1	-2	+1	-1	+1	+1	+1	0	0	0	0	0	0	0	0	= **+3**

Tabela 3.3-7 – Alinhamento semiglobal (1) entre as seqüências *s*=CAGCACTTGGATTCTCGG e *t*=CAGCGTGG.

Ou

	M	M	M	M	T	T	T	T	T	M	T	M	T	T	T	T	M	M	
s'	**C**	**A**	**G**	**C**	**A**	**C**	**T**	**T**	**G**	**G**	**A**	**T**	**T**	**C**	**T**	**C**	**G**	**G**	
t'	**C**	**A**	**G**	**C**	**-**	**-**	**-**	**-**	**-**	**G**	**-**	**T**	**-**	**-**	**-**	**-**	**G**	**G**	
Escore	+1	+1	+1	+1	-2	-2	-2	-2	-2	+1	-2	+1	-2	-2	-2	-2	+1	+1	= **-12**

Tabela 3.3-8 – Alinhamento semiglobal (2) entre as seqüências *s*=CAGCACTTGGATTCTCGG e *t*=CAGCGTGG.

Com relação a todos os métodos de alinhamento acima, ilustrativamente foi usado um escore bastante simplificado e que até pode ser adotado para seqüências de DNA, mas uma estratégia mais elaborada é necessária para comparações de seqüências de proteína, como será visto mais adiante pelo uso de matrizes de substituição, as quais levam em conta os eventos mutacionais que causam as chamadas substituições conservativas.

Nas buscas em bancos de dados de proteínas, é necessário lembrar-se de que a habilidade de um método em reconhecer seqüências distantemente relacionadas (sensitividade) é tão importante quanto evitar aquelas não-relacionadas mesmo com altos escores de similaridade (seletividade), pois, com certeza, existem mais do último tipo do que do primeiro e isso tem efeitos dramáticos sobre a qualidade e a *performance* da

busca. Calcula-se que existam mais de 500.000 seqüências de proteínas já armazenadas em bancos de dados, enquanto uma típica família de proteínas tem menos de 100 membros. Assim, algoritmos de comparação, matrizes de substituição e penalidades de espaços que simplesmente produzam o melhor alinhamento não são mais apropriados para buscas em bancos de dados **[Pearson, 2001]**.

Conforme alerta **[Gusfield, 1997]**, a utilidade de um alinhamento é afetada pelo esquema de escore utilizado, o que é bastante perceptível principalmente para os tipos local e global. Por exemplo, se às identidades é atribuído o valor 1, e às divergências e aos espaços, zero, então, o alinhamento local ótimo será determinado pela subseqüência comum mais longa. Contrariamente, se às divergências e aos espaços forem atribuídos valores negativos altos, e à cada identidade, o valor 1, então o alinhamento local ótimo será a subsérie comum mais longa. Pode ser que na maioria dos casos, nenhum desses seja o alinhamento de interesse, haja vista que o escore depende também da característica da aplicação. Para alinhamento local, as entradas na matriz de substituição devem ter um escore médio que seja negativo. De outro modo, o alinhamento resultante tenderá a ser global. No contexto de busca em bancos de dados, novas teorias vêm sendo desenvolvidas, dentre elas o escore *log-odds* (vide **[Gusfield, 1997, p. 389]**).

Se a comparação de seqüência é feita para testar a hipótese de origem comum, deve-se tomar cuidado quanto à interpretação dos resultados. Em geral, alinhamentos ótimos são os mais improváveis de terem ocorrido *"por acaso"*[50], probabilisticamente analisando. Entretanto, é sempre prudente comparar os escores obtidos àquele que seria esperado em média para seqüências não relacionadas, mas com as mesmas características das seqüências em questão. Se um escore ótimo está bem acima da média, essa é uma boa indicação de que a similaridade entre as seqüências não é por acaso. Mesmo assim, esse resultado por si mesmo não implica homologia ou qualquer tipo de evidência de origem comum. Experimentos mais detalhados, baseados na informação que o alinhamento provê, são em geral realizados para fortalecer ou refutar a hipótese de ancestral comum. Por outro lado, se a similaridade está muito próxima (ou igual) àquela que é esperada *"por acaso"*, é provável que as seqüências não sejam mesmo relacionadas. Entretanto, em biologia não há regras sem exceções e há casos de proteínas homólogas sem qualquer traço de similaridade **[Meidanis e Setubal, 1997]**.

3.3.3 A importância evolutiva dos *gaps*[51] e suas funções de penalidade

Até o momento, as idéias centrais usadas para medir o valor de um alinhamento (sua similaridade) foram identidades, divergências e espaços. No entanto, já é o momento de se mencionar um outro conceito : lacuna ou *gap*[xviii]. **[Meidanis e Setubal, 1997]** Uma lacuna pode ser definida como sendo um conjunto máximo e consecutivo de $k > 1$ espaços que ocorrem numa seqüência quando se tenta alinhá-la a outra, ou a outras. **[Gusfield, 1997]** As lacunas ajudam a criar alinhamentos que melhor se conformam aos modelos biológicos e se ajustam bem às buscas de padrões. Um exemplo dessa definição é mostrado na Tabela 3.3-9 abaixo, a qual contém duas lacunas e dois espaços isolados:

S_1	C	G	G	T	A	A	C	-	-	A	-	A	C
S_2	C	-	-	-	G	A	C	G	C	A	T	-	C

Tabela 3.3-9 – Um exemplo de alinhamento *pairwise* mostrando duas lacunas e dois espaços isolados.

É geralmente aceito que uma lacuna com *k* espaços seja mais provável que a mera ocorrência de *k* espaços isolados, pois uma lacuna pode acontecer devido a um evento mutacional que cause a remoção ou inserção de uma subseqüência inteira (particularmente em DNA), enquanto espaços separados acontecem mais provavelmente devido a eventos distintos, e esse acontecimento é menos comum que as lacunas **[Meidanis e Setubal, 1997]**. Alguns exemplos ilustrativos dessas situações são **[Gusfield, 1997]**: *crossover desigual*[52] (*unequal crossing-over*) em meiose[53], que causa uma inserção em uma seqüência e uma exclusão recíproca na outra; *DNA slippage* durante a replicação (onde uma porção do DNA é repetida na cópia replicada porque a máquina de replicação se perdeu na fita-modelo, escorregando para trás e repetindo uma seção); inserção de elementos transponíveis (*jumping* genes) numa seqüência de DNA; inserções de DNA por *retrovírus*[54]; translocações [55] de DNA entre cromossomos; o alinhamento de um DNA genômico a um cDNA, onde os gaps corresponderão à ausência dos íntrons; alinhamento de proteínas, em que as seqüências podem ser bastante similares em vários intervalos, mas diferirem em outros que contenham domínios específicos, causando as lacunas.

Um alinhamento de duas seqüências reflete o custo (ou a probabilidade) de eventos mutacionais necessários para transformar uma seqüência em outra. Posto que uma lacuna de mais de um espaço pode ser criada por um único evento, o modelo de

[xviii] O termo em inglês já faz parte do vocabulário técnico de bioinformática e, por isso, será usado de agora em diante.

alinhamento deve, sempre que possível, refletir o verdadeiro significado biológico da distribuição dos espaços nas lacunas, não meramente o número de espaços no alinhamento, pois o modo como os valores são atribuídos influencia criticamente a efetividade do conceito. Não existem muitas funções de penalidade de lacunas e somente algumas levam em conta essa necessidade. Atualmente as principais funções (ou modelos) adotados são **[Gusfield, 1997]**:

a) Constante: é a escolha mais simples, onde cada espaço individual é livre e um valor V_g é dado a cada lacuna, independente da quantidade de espaços que contenha. Sendo V_i e V_d os valores para identidades e divergências, respectivamente, encontrar um alinhamento máximo poderia ser da forma $[V_i(\#\ identidades) - V_d(\#\ divergências) - V_g(\#\ lacunas\ ou\ gaps)]$. Mais genericamente, se os valores de identidades e divergências forem dependentes de alfabeto (DNA, proteína etc), então o objetivo do modelo será encontrar um alinhamento A que maximize $(\Sigma^{l}_{i=1}[s(S'_1(i),S'_2(i))] - V_g(\#\ lacunas\ ou\ gaps))$, onde $s(x,-) = s(-,x) = 0$ para cada caracter x porque os espaços isolados não são considerados, e S'_1 e S'_2 representam as seqüências S_1 e S_2 após a inserção de espaços.

b) Afim ou linear: é provavelmente o modelo mais utilizado na literatura de biologia molecular e pode ser considerado uma generalização do modelo constante, sendo que aqui é adicionado um valor V_e para cada espaço na lacuna. Assim, o V_g é chamado de valor de iniciação da lacuna, pois representa o custo de iniciar a lacuna; e o V_e, valor de extensão da lacuna porque indica o custo de cada espaço que a compõe. Então, encontrar um alinhamento máximo poderia ser da forma $[V_i(\#\ identidades) - V_d(\#\ divergências) - V_g(\#\ lacunas\ ou\ gaps) - V_e(\#\ espaços)]$. O valor para uma única lacuna de tamanho q é dado pela função $V_g + qV_e$. A versão dependente de alfabeto, de novo, atribui $s(x,-) = s(-,x) = 0$ e tem o objetivo de encontrar um alinhamento A que maximize $(\Sigma^{l}_{i=1}[s(S'_1(i),S'_2(i))] - V_g(\#\ lacunas\ ou\ gaps) - V_e(\#\ espaços))$. É comum usar um valor alto para V_g (entre 10 e 15, por exemplo no contexto da matriz BLOSUM62) e um valor baixo para V_e (entre 1 e 2).

c) **Convexa ou côncava**: alguns fenômenos biológicos são mais bem modelados por uma função de penalidade de lacuna quando nessa o valor de um espaço adicional contribui menos que aquele do espaço precedente. Um exemplo seria a função $V_g + log_e q$, onde q é o tamanho da lacuna.
d) **Arbitrária**: é o tipo mais genérico, onde o valor de uma lacuna é uma função arbitrária $v(q)$ do seu tamanho q. Todos os modelos anteriores são subcasos deste.

3.3.4 Algoritmos

Um enfoque para computar a similaridade entre duas seqüências seria gerar todos os possíveis alinhamentos e então considerar apenas o melhor deles. Entretanto, o número de alinhamentos entre duas seqüências é exponencial, e tal enfoque resultaria um algoritmo intoleravelmente lento **[Meidanis e Setubal, 1997]**.

Por isso, duas classes genéricas de algoritmos são usadas para calcular os escores de similaridade quando se deseja inferir homologia: algoritmos rigorosos que garantem o cálculo de um escore ótimo pela aplicação da estratégia chamada *programação dinâmica* , isto é, os algoritmos Needleman-Wunsch (1970) e Smith-Waterman (1981), sendo o primeiro aplicável para alinhamentos globais e o último, para alinhamentos locais; e os algoritmos heurísticos rápidos BLAST e FASTA, que não garantem um escore ótimo para cada seqüência numa base de dados **[Pearson, 2001]**.

3.3.5 Algoritmos de Programação Dinâmica

O embasamento da programação dinâmica vem da observação de que qualquer subcaminho parcial que termine em um ponto ao longo do caminho ótimo principal deve ele mesmo ser o caminho ótimo que leva àquele ponto. Assim, o caminho ótimo pode ser encontrado por extensão incremental de subcaminhos ótimos. Na formulação básica de Needleman-Wunsch, o alinhamento ótimo deve se estender do começo ao fim das duas seqüências, ou seja, do canto superior esquerdo no espaço de busca ao canto inferior direito. Em outras palavras, ele busca alinhamentos globais. Uma simples modificação dessa estratégia permite que um alinhamento local ótimo seja encontrado pelo algoritmo Smith-Waterman. O caminho para esse alinhamento não precisa atingir as margens do

gráfico de busca, mas pode começar e terminar internamente a esse. Tal alinhamento seria localmente ótimo se seu escore não pudesse ser melhorado pelo aumento ou pelo decréscimo de sua extensão. Esse algoritmo está embasado por uma propriedade de que o escore acumulativo para um caminho decrescerá em regiões de seqüências com baixa identidade. Quando o escore cai a zero, a extensão do caminho é terminada e um novo caminho pode ser iniciado. Podem existir muitos caminhos individuais limitados por regiões de baixa identidade; e aquele com o mais alto escore será, então, relatado como o alinhamento local ótimo **[Baxevanis e Ouellette, 2001]**.

É importante ter em mente que métodos ótimos sempre reportam o melhor alinhamento que pode ser encontrado, mesmo que não apresente nenhum significado biológico. Por outro lado, quando se busca por alinhamentos locais, pode ser que existam vários alinhamentos significativos, e seria um erro considerar apenas aquele ótimo. Refinamentos ao algoritmo Smith-Waterman foram propostos para detectar os *k* melhores alinhamentos locais. Essas idéias foram mais tarde estendidas no desenvolvimento do algoritmo SIM (por Huang em 1990) e um programa chamado *lalign* (distribuído com o pacote FASTA) provê uma implementação útil desse algoritmo. A busca por subalinhamentos é especialmente importante quando proteínas multimodulares são comparadas.

A programação dinâmica consiste, então, em resolver uma instância de um problema a partir de soluções já computadas para instâncias menores do mesmo problema. Dadas duas seqüências *s* e *t*, constróem-se as soluções pela determinação de todas as similaridades entre prefixos arbitrários das duas seqüências. Inicia-se com prefixos menores e usam-se resultados já computados para resolver o problema para prefixos maiores **[Meidanis e Setubal, 1997]**.

Sejam *m* o tamanho de *s* e *n* o tamanho de *t*. Existirão *m*+1 prefixos possíveis de *s* e *n*+1 prefixos possíveis de *t*, incluíndo a seqüência vazia. Assim, ter-se-á uma matriz de (*m*+1) x (*n*+1) elementos, onde cada entrada (*i*, *j*) contém a similaridade entre *s*[1..*i*] e *t* [1..*j*]. Sendo *s*=AAAC e *t*=AGC, a primeira seqüência é colocada na margem esquerda e a segunda, no topo, para que os prefixos sejam mais facilmente indicados. **[Gusfield, 1997]** Para resolver o problema, são utilizados três componentes essenciais: a relação recorrente, a computação tabular e o rastreamento reverso. A ênfase deste tópico está

direcionada para o alinhamento global, mas, sempre que necessário, alguns detalhes do alinhamento local são adicionados, visando a utilizar estratégias semelhantes para os dois processos.

A relação recorrente estabelece uma recursividade entre o valor da entrada *a*(*i*, *j*), para *i* e *j* positivos, e os valores de entrada com pares de índices menores que *i* e *j*. Quando não existirem índices menores, o valor de *a*(*i*, *j*) deve ser determinado explicitamente pela chamada condição-base: ***a(i, 0) = -2i*** e ***a(0, j) = -2j***, onde *–2* é a penalidade para cada espaço. Isso porque existe apenas um alinhamento possível se uma das seqüências for vazia: adicionam-se tantos espaços quanto forem os caracteres na outra seqüência. O escore desse alinhamento será *–2k*, onde *k* é o tamanho da seqüência não vazia. Para alinhamento local, a condição-base seria atribuir o valor zero para todas as células da primeira linha e também da primeira coluna, da forma : ***a(i, 0) = 0*** e ***a(0, j) = 0.*** Logo em seguida, são apresentadas as tabelas da condição-base tanto para o alinhamento global quanto para o alinhamento local (Figuras 3.3-2 e 3.3-3) **[Meidanis e Setubal, 1997]**.

Para computar os demais valores, a observação-chave é que se pode computar o valor para *a*(*i*, *j*) a partir de três entradas prévias: (*i*, *j* - 1), (*i* - 1, *j* - 1) e (*i* - 1, *j*). **[Meidanis e Setubal, 1997]** A razão é que existem apenas três caminhos exaustivos para obter um alinhamento entre *s*[1..*i*] e *t* [1..*j*], pois não se pode ter dois espaços pareados na última coluna do alinhamento:

- alinhar *s*[1..*i*] com *t* [1..*j – 1*], o que significa parear um espaço com *t[j]*, ou
- alinhar *s*[1..*i - 1*] com *t* [1..*j – 1*], pareando s*[i]* com *t[j]*, ou
- alinhar *s*[1..*i - 1*] com *t* [1..*j*], o que produz o par s*[i]* com um espaço.

Os escores dos melhores alinhamentos entre os prefixos menores ficam armazenados na matriz se uma ordem apropriada é escolhida para computar as entradas. Como conseqüência, a similaridade sim(*s*[1..*i*], *t* [1..*j*]) da matriz *a* pode ser determinada pela fórmula a seguir (Figura 3.3-1), na qual *p(i, j)* é 1 se s*[i]=t[j]* e –1 se s*[i]* ≠ *t[j]* :

$$a[i, j] = \max \begin{cases} a[i, j-1] - 2 \\ a[i-1, j-1] + p(i, j) \\ a[i-1, j] - 2 \end{cases}$$

Figura 3.3-1 – A fórmula da relação recorrente de programação dinâmica.

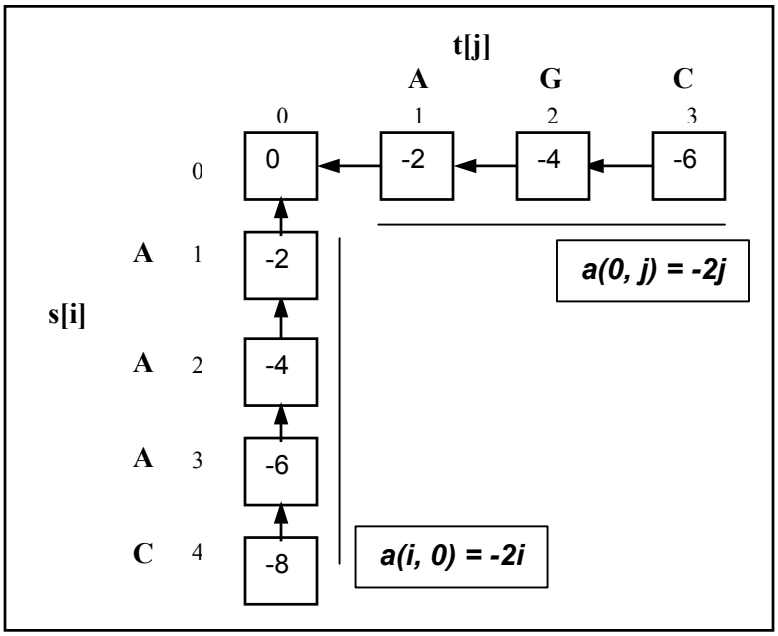

Figura 3.3-2 – A condição-base da relação recorrente do alinhamento global para as seqüências s=AAAC e t=AGC.

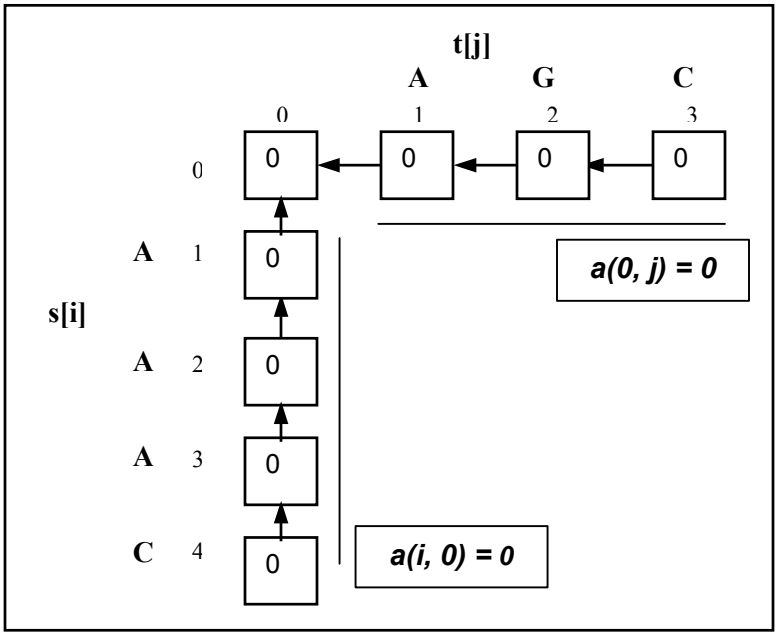

Figura 3.3-3 – A condição-base da relação recorrente do alinhamento local para as seqüências s=AAAC e t=AGC.

A computação tabular (ou computação da tabela) é o segundo componente essencial da programação dinâmica e usa as relações recorrentes para eficientemente computar o valor da entrada *a[i, j]* **[Gusfield, 1997]**. Essa tarefa pode ser implementada usando qualquer linguagem de programação que permita recursão, por exemplo Java. Por um enfoque *bottom-up* dos menores para os maiores valores de *i* e *j*, a matriz é preenchida linha a linha (da esquerda para a direita) ou coluna por coluna (de cima para baixo). Para computar, por exemplo, o valor de *a[1,1]*, os três valores prévios necessários seriam:

- *a[1, 0] –2 = – 2 –2 = – 4*, sendo o segundo valor –2 relativo à penalidade de espaço ;
- *a[0, 0] + 1 = 0 + 1 = 1*, pois *s[1]* é igual a *t[1]*, resultando *p(i, j)* igual a 1;
- *a[0, 1] –2 = – 2 –2 = – 4*, sendo o segundo valor –2 também relativo à penalidade de espaço.

Dentre os valores encontrados, a *a[i,j]* será sempre atribuído o maior (ou o máximo) deles. Para o exemplo acima, *a[1,1]* será igual a 1. O mesmo se repete para preencher os valores das demais células, utilizando o algoritmo de alinhamento global proposto por **[Meidanis e Setubal, 1997]** (Figura 3.3-4), ou o algoritmo apresentado por **[Pearson, 2001, p.27]** para o alinhamento local (Figura 3.3-5):

```
Algoritmo Similaridade Global
    entrada: seqüências s e t,  saída: similaridade entre s e t
    m ← |s|, n ← |t|
    for i ← 0 to m do     //condição-base para a coluna 0
        a[i , 0] ← i * g    //g é a penalidade para a ocorrência de cada espaço
    for j ← 0 to n do     // condição-base para a linha 0
        a[0, j] ← j * g
    for i ← 1 to m do     //a relação recorrente para as células i(1..m) e j(1..n)
        for j ← 1 to n do
            a[i, j] ← max( a[i, j – 1] + g,  a[i – 1, j – 1] + p(i, j),  a[i – 1, j] + g )
    return a[m, n]                  //retorna a matriz preenchida
```

Figura 3.3-4 – Um algoritmo (pseudocódigo) para calcular a similaridade global em programação dinâmica.

```
Algoritmo Similaridade Local
    entrada: seqüências s e t,  saída: similaridade entre s e t
    m ← |s|, n ← |t|, melhorEscore ← 0
    for i ← 0 to m do        //condição-base para a coluna 0
        a[i , 0] ← 0         //o valor de cada célula é zero
    for j ← 0 to n do        // condição-base para a linha 0
        a[0, j] ← 0          //o valor de cada célula também é zero
    for i ← 1 to m do        //a relação recorrente para as células i(1..m) e j(1..n)
        for j ← 1 to n do
            a[i, j] ← max(0,  a[i, j – 1] + g ,   a[i – 1, j – 1] + p(i, j),   a[i – 1, j] + g )
            melhorEscore ← max( a[i, j], melhorEscore ) //o melhor escore
    return a[m, n]           //retorna a matriz preenchida
```

Figura 3.3-5 – Um algoritmo (pseudocódigo) para calcular a similaridade local em programação dinâmica.

Esquematicamente, os valores calculados passam a ocupar uma determinada posição em cada célula, sendo que o valor *a[i,j]* fica posicionado no centro e o valor de *p(i, j),* no canto superior esquerdo. Uma seta é posicionada na direção da célula ou das células que deram origem ao valor *a[i,j]* máximo, conforme Figuras 3.3-6 e 3.3-7 a seguir, baseadas em **[Meidanis e Setubal, 1997]**:

[Vide próxima página]

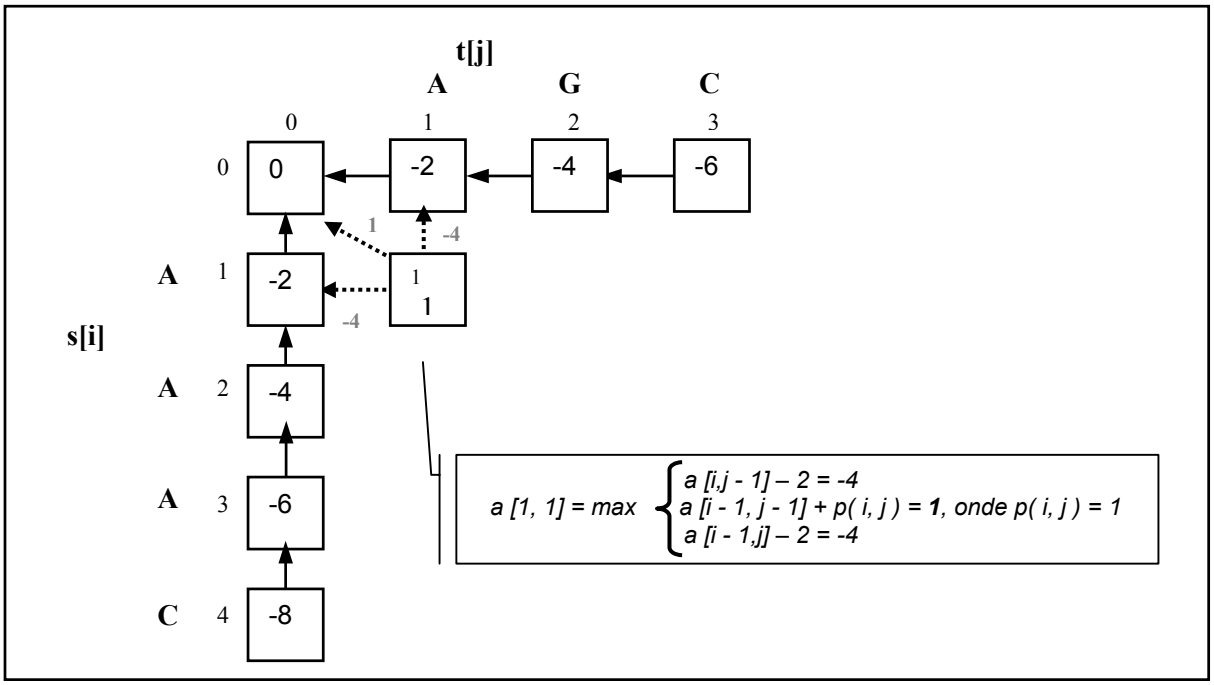

Figura 3.3-6 – O início da computação tabular para <u>alinhamento global</u> das seqüências s=AAAC e t=AGC (parte 1 de 2)**.**

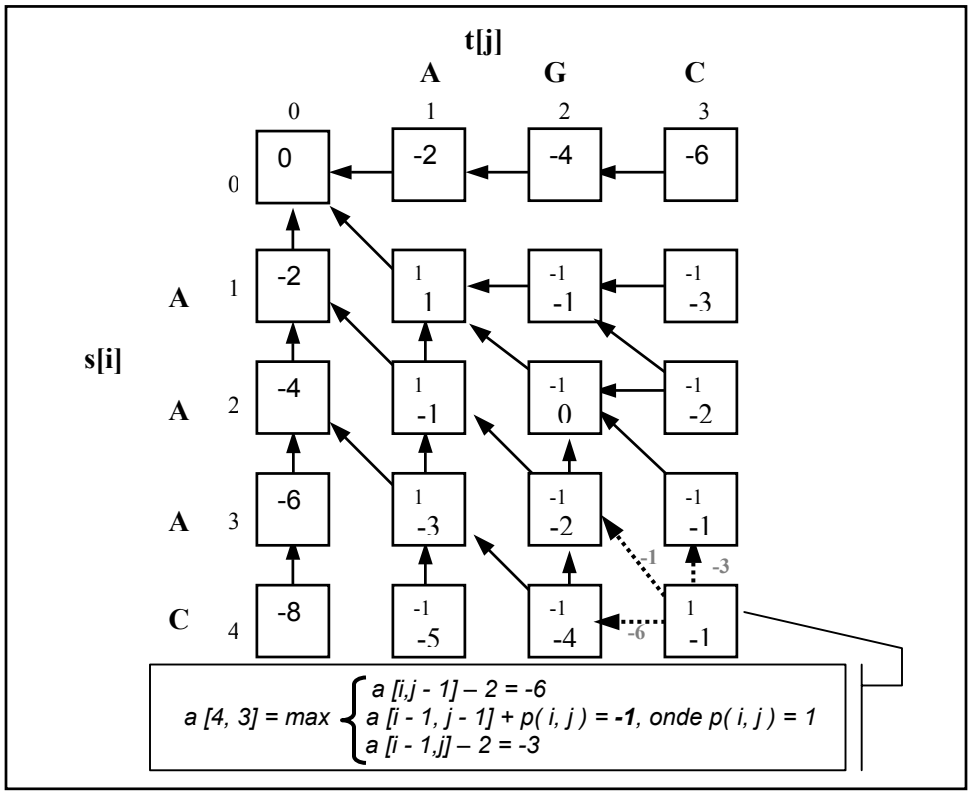

Figura 3.3-7 – O resultado final da computação tabular para <u>alinhamento global</u> das seqüências s=AAAC e t=AGC (parte 2 de 2)**.**

Para o alinhamento local, as tabelas armazenariam valores diferentes daqueles do alinhamento global, apesar de a estratégia ser a mesma (Figuras 3.3-8 e 3.3-9):

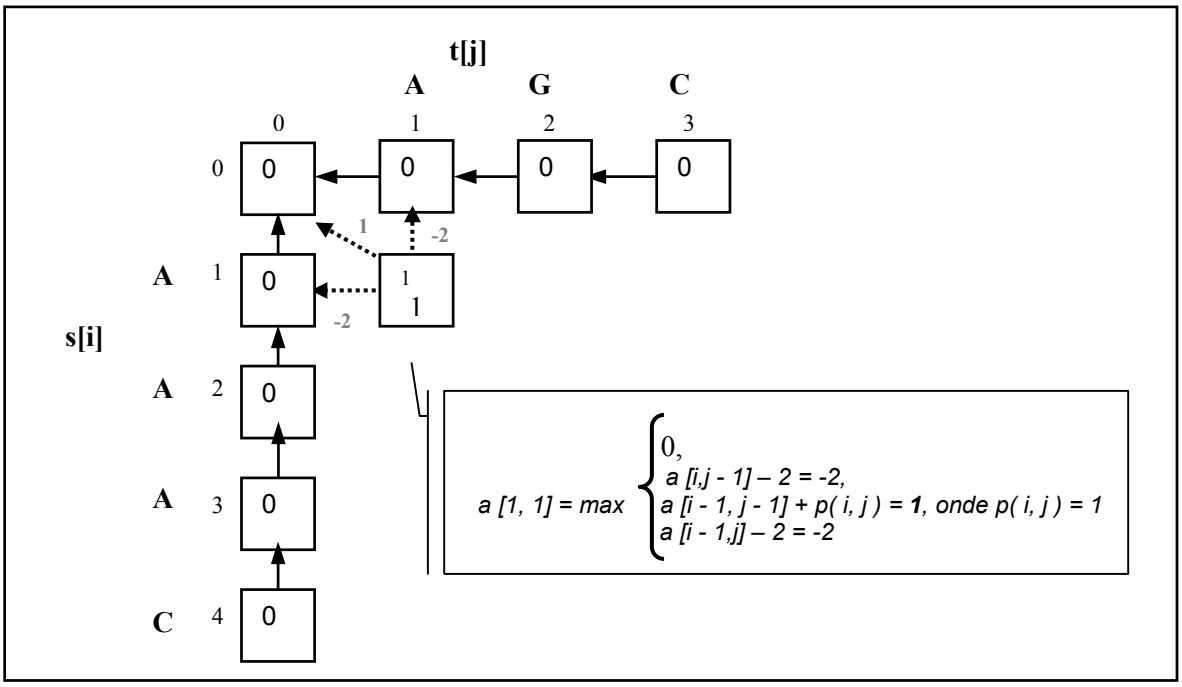

Figura 3.3-8 – O início da computação tabular para alinhamento local das seqüências s=AAAC e t=AGC (parte 1/ 2)**.**

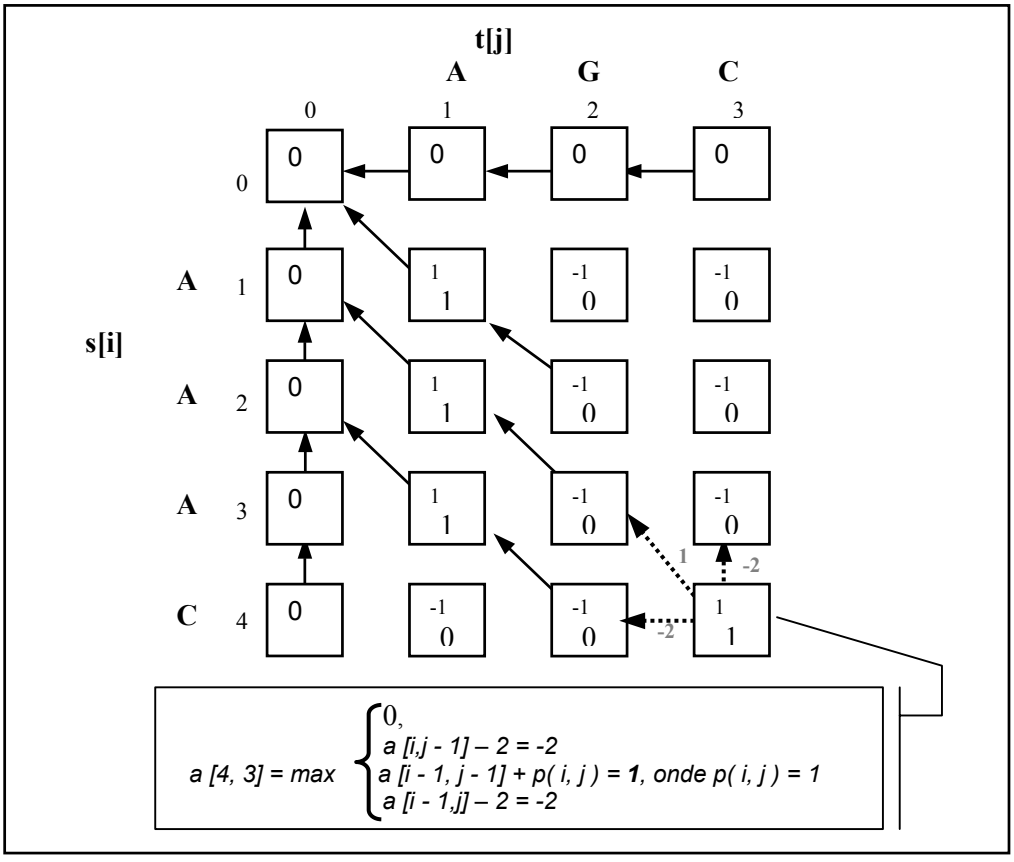

Figura 3.3-9 – O resultado final da computação tabular para alinhamento local das seqüências s=AAAC e t=AGC (parte 2 / 2)**.**

O terceiro e último componente essencial da programação dinâmica é o rastreamento reverso, por meio do qual se podem construir todos os possíveis alinhamentos (local e global) entre as duas seqüências. Da mesma maneira que se procedeu quanto à computação tabular, o rastreamento reverso também pode ser executado por um algoritmo recursivo.

Em seguida, é apresentado o pseudocódigo do algoritmo *Align(m, n, len)*, o qual construirá o caminho para o alinhamento ótimo, tendo como parâmetros a matriz a e as seqüências s e t (Figura 3.3-10). Os caracteres alinhados, que tanto podem ser espaços ou símbolos, estarão armazenados nas posições *1...len* (última posição ou tamanho) dos vetores *align-s* e *align-t*. O escopo (visibilidade) desses vetores é global. O retorno desse algoritmo é o atributo tamanho do alinhamento. É preciso notar que $max(|s|, |t|) \leq len \leq m + n$, ou seja, o tamanho do alinhamento é maior ou igual ao valor máximo entre o tamanho das seqüências *s* ou *t* e, no entanto, é menor igual à soma desses tamanhos [**Meidanis e Setubal, 1997**].

```
Algoritmo Align (Alinhamento)
    entrada     : índices i, j e a matriz a do algoritmo Similaridade.
    saída       : alinhamento nos vetores align-s e align-t, e o tamanho do alinhamento( len ).
    if i = 0 and j = 0 then
        len ← 0
    else if i > 0 and a[i, j] = a[i - 1, j] + g then
        Align(i - 1, j, len)
        len ← len + 1
        align – s[len] ← s[i]
        align – t[len] ← ' – '
    else if i > 0 and j > 0 and = a[i, j] = a[i - 1, j - 1]+ p(i, j) then
        Align(i - 1, j - 1, len)
        len ← len + 1
        align – s[len] ← s[i]
        align – t[len] ← t[j]
    else  // deve ser j > 0 e a[i, j] = a[i, j - 1] + g
        Align(i, j - 1, len)
        len ← len + 1
        align – s[len] ← ' – '
        align – t[len] ← t[j]
```

Figura 3.3-10 – Um exemplo de algoritmo (pseudocódigo) para calcular o alinhamento ótimo em programação dinâmica.

Dentre outros autores, **[Baxevanis e Ouellette, 2001]** mencionam que podem existir diversos alinhamentos para um mesmo par de seqüências. No entanto, o algoritmo Align (Figura 3.3-10), extraído de **[Meidanis e Setubal, 1997]**, retorna apenas um deles pela preferência anti-horária das setas na ordem vertical, diagonal e horizontal, conforme se observa na Figura 3.3-11:

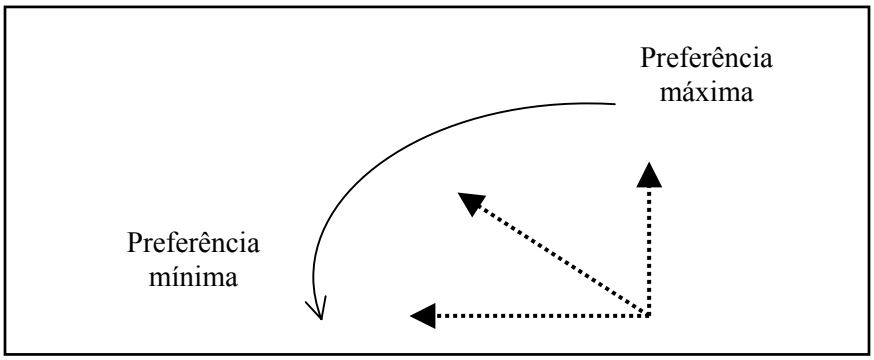

Figura 3.3-11 – Preferência anti-horária das setas no rastreamento reverso.

Dessa forma, o alinhamento ótimo retornado pelo algoritmo tem as seguintes características: quando houver escolha, uma coluna com um espaço na seqüência *t* tem precedência sobre uma coluna com dois símbolos, a qual, por sua vez, tem precedência sobre uma coluna com um espaço em *s*. Por isso, o alinhamento global ótimo rastreado para as seqüências *s*=AAAC e *t*=AGC é (Tabela 3.3-10):

s'	**A**	**A**	**A**	**C**	
t'	**A**	**G**	**-**	**C**	
Escore	**1**	**-1**	**-2**	**1**	= -1

e não, por exemplo,

s'	**A**	**A**	**A**	**C**	
t'	**-**	**A**	**G**	**C**	
Escore	**-2**	**1**	**-1**	**1**	= -1

Tabela 3.3-10 – Alinhamento global ótimo para as seqüências *s*=AAAC e *t*=AGC.

Quanto ao escore, mostrado na última linha das Tabelas 3.3-10, este pode ser encontrado sempre na última célula do alinhamento, ou na última célula da matriz quando o alinhamento se estender de um extremo a outro, que é o caso das seqüências aqui consideradas.

Visto sob o aspecto gráfico, o rastreamento reverso também utiliza a tabela preenchida pela computação tabular e construirá o alinhamento ótimo mostrado acima. Então, a partir da tabela (Figura 3.3-12), tudo que se precisa fazer é tomar um caminho iniciando na posição *(m, n)* e seguir as setas até que se chegue à posição *(0,0)*. **[Meidanis**

e Setubal, 1997] Cada seta representa uma das três possíveis colunas do alinhamento: se essa seta for horizontal, ela corresponderá a uma coluna com um espaço em *s* pareado com *t[j],* o que pode ser interpretado como uma *inserção* do caracter *t[j]* em *s* ; se for vertical, corresponderá a *s[i]* pareado com um espaço em *t*, sendo interpretada como uma *exclusão* do caracter *s[i]* com relação a *t*; e, finalmente, uma seta diagonal significa *s[i]* pareado com *t[j],* que pode ser interpretada como uma *substituição* quando *s[i] ≠ t[j]*.

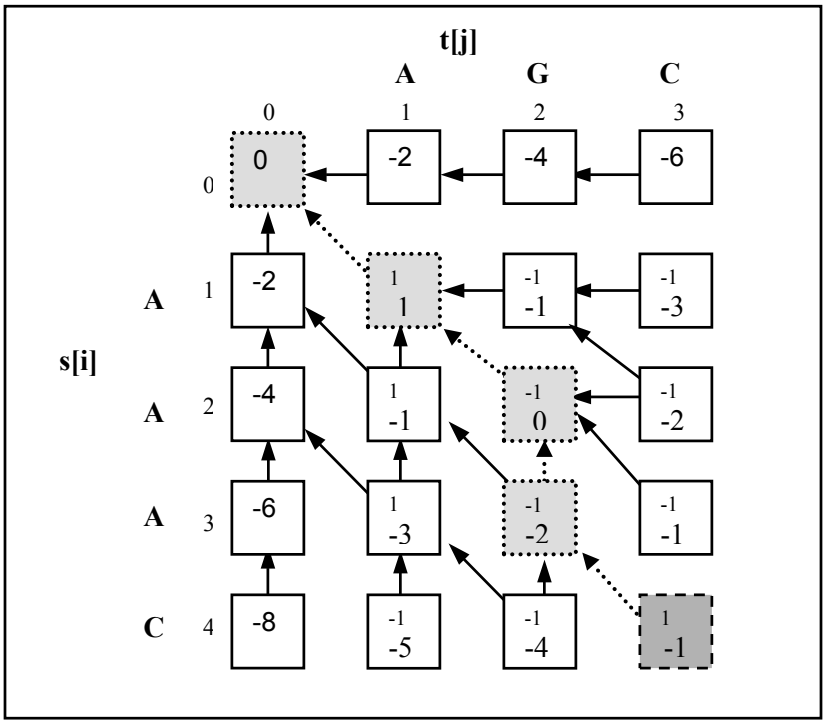

Figura 3.3-12 – O alinhamento global ótimo em programação dinâmica para as seqüências s=AAAC e t=AGC.
De baixo para cima, o valor do escore é = +1(igualdade na diagonal) -2 (gap em t na vertical) -1(desigualdade na diagonal) +1(igualdade na diagonal) = **–1**, armazenado na **última célula (4,3)** da matriz.

Quanto ao alinhamento local, o caminho ótimo do pareamento entre as subseqüências *s'*=AAC e *t'*=AGC se estende da posição (1,0) até a última célula, o que pode ser visto na Tabela 3.3-11 e na Figura 3.3-13 a seguir:

s'	**A**	**A**	**A**	**C**	
t'	**-**	**A**	**G**	**C**	
Escore		**1**	**-1**	**1**	**= 1**

Tabela 3.3-11 – Alinhamento global ótimo para as seqüências *s*=AAAC e *t*=AGC.
(1ª. parte / 2)

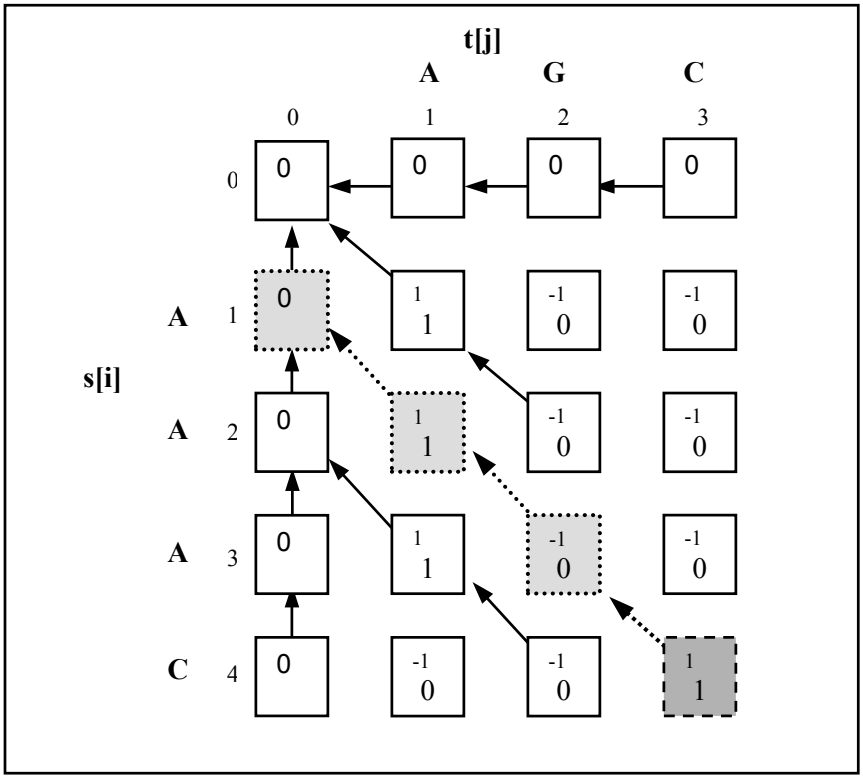

Figura 3.3-13 – O alinhamento local ótimo em programação dinâmica para as seqüências s=AAAC e t=AGC.
O valor do escore é = 2(+1) +1(-1) = **1**, armazenado na **última célula (4,3)** da matriz. (2ª. parte / 2)

Quanto à complexidade, os algoritmos de programação dinâmica utilizados neste tutorial podem ser assim analisados **[Meidanis e Setubal, 1997]**:

- Similaridade: os dois algoritmos apresentam complexidade *O(mn)* tanto para o tempo dispendido quanto para o espaço utilizado, pois o número de operações que executam depende essencialmente do número de entradas da matriz *(m x n)*. Em cada um dos usos da estrutura de repetição **for**, pode-se perceber que sempre se está considerando toda a extensão da primeira ou da segunda seqüências (*s* ou *t*). Se as seqüências tiverem tamanho igual ou aproximado, então a complexidade passa a ser $O(n^2)$ ou, mais comumente, complexidade quadrática.
- Alinhamento: a construção do alinhamento ótimo, pelo fato de a matriz já estar devidamente preenchida, é feito em um tempo *O(len)*, onde *len* é o tamanho do alinhamento retornado, o qual é *O(m+n)*.

3.3.6 Matrizes de substituição ou de escore

Matrizes de substituição, ou por vezes também chamadas de matrizes de escore, são tabelas bidimensionais (i, j) que contêm valores que demonstram a probabilidade de que o aminoácido da posição i sofra mutação para o aminoácido da posição j, quaisquer que sejam as duas seqüências envolvidas. Tais matrizes são construídas pelo estudo de diversas amostras de alinhamento *pairwise*. Se a amostra é grande o suficiente para ser estatisticamente significante, as matrizes devem refletir as verdadeiras possibilidades de mutações que ocorreram ao longo de um certo período de evolução. Dois enfoques gerais têm sido usados para produzir tais matrizes: a) o primeiro, um modelo pioneiro de evolução de proteínas, que resultou num conjunto de matrizes denominadas PAM (***P****oint* ***A****ccepted* ***M****utation* ou Mutação Pontual Aceita); e b) um outro modelo mais recente chamado BLOSUM (***Blo****ck* ***Su****bstitution* ***M****atrices* ou Matrizes de Substituição de Blocos) **[NCBI/Education, 2002]**.

Os alinhamentos computados para buscas em bancos de dados de proteínas são quase sempre valorados, e a escolha da matriz de escore pode ter um efeito considerável sobre os resultados. Sugere-se, em alguns casos, que essa escolha é o elemento técnico mais crítico para o sucesso de uma busca. Idealmente, os valores na matriz devem refletir os fenômenos biológicos que os alinhamentos procuram mostrar, por exemplo: no caso de divergência de seqüências por mutações evolucionárias, os escores devem preferencialmente ser derivados de observações empíricas entre seqüências ancestrais e seus atuais descendentes; já no caso de motivos conservados ou correlações bem definidas entre estrutura e seqüência, os números devem ser derivados de coleções de seqüências contendo esses padrões desejados **[Gusfield, 1997]**.

O sistema de escore descrito até o momento utiliza um esquema bastante simplificado: +1 para identidade, -1 para divergência e –2 para espaço. No entanto, ao se compararem seqüências de proteínas, essa estratégia não é suficiente, principalmente nos casos de alinhamentos de baixos escores por pouca sensitividade[56] do algoritmo. Isso porque, num cenário evolutivo, as mutações não ocorrem de forma assim tão primitiva. **[Baxevanis e Ouellette, 2001]** Sabe-se que certos aminoácidos podem ser conservativamente substituídos por outros, presumivelmente devido à similaridade de suas propriedades físico-químicas, por exemplo: isoleucina por valina (ambos pequenos e hidrofóbicos) e

serina por treonina (ambos polares). Outras propriedades, tais como tendência de se ligar a moléculas de água (hidrofilia), também influenciam a probabilidade de substituição mútua.

Devido a essas características das comparações de proteínas, é importante usar um esquema de escore que reflita as probabilidades tanto quanto possível: as identidades devem receber escores maiores que as substituições, sendo que às conservativas devem-se atribuir valores mais significativos que às não-conservativas. **[Baxevanis e Ouellette, 2001]** Além do mais, diferentes conjuntos de valores podem ser requeridos para comparar seqüências muito similares (gene do camundongo e seu homólogo ratazana) em oposição àquelas altamente divergentes (genes de camundongo e levedura). Essas considerações podem ser tratadas de uma maneira mais flexível pelo uso de uma matriz de substituição, na qual o escore de um par aleatório de aminoácidos pode ser facilmente encontrado.

As matrizes mais efetivas são baseadas em uma observação direta das frequências atuais de substituição dos diversos pares de resíduos entre proteínas relacionadas, sendo esse caminho apontado como o melhor para derivar escores de similaridades**. [Pearson, 2001]** Por isso, as matrizes de escores podem diferir de três maneiras: a) o método pelo qual são construídas; b) o conteúdo da informação, o qual está relacionado o número de resíduos que deve ser alinhado para produzir um escore estatísticamente significante; c) a escala, isto é, a quantidade de informação provida por unidade de escore.

3.3.7 PAM

O modelo PAM de evolução de proteínas é também chamado de MDM (***M**utation **D**ata **M**atrices* ou Matrizes de Dados de Mutação) ou Matrizes de Dayhoff. Foi desenvolvido em 1978 por um projeto pioneiro da Fundação Nacional de Pesquisa Biomédica dos Estados Unidos (NBRF) e liderado pela pesquisadora Margaret Oakley Dayhoff.

Margaret e sua equipe fizeram um estudo abrangente das frequências nas quais os aminoácidos se substituem uns pelos outros durante a evolução, baseando-se em três hipóteses: a) os eventos mutacionais são independentes do contexto; b) um acontecimento mutacional numa certa posição é independente dos eventos mutacionais

anteriores que tiveram lugar nessa posição; c) a probabilidade de substituição de X por Y é a mesma de Y por X. **[PSC, 1999]** Esses estudos envolveram alinhamentos globais de 1572 proteínas de 71 famílias relacionadas com, pelo menos, 85% de similaridade e, em seguida, a construção de árvores filogênicas para cada uma dessas famílias. Cada árvore foi examinada pelas substituições encontradas em cada galho/ramo (par de seqüência). Essas freqüências relativas foram colocadas numa matriz 20 x 20 representando todas as possíveis combinações de substituições entre os aminoácidos. A matriz foi, então, normalizada para valores que representassem a probabilidade de que 1% (1 em 100) dos aminoácidos viesse a experimentar uma mutação, resultando na matriz PAM1 **[Rocha, 2000]**.

De maneira mais detalhada, **[Gusfield, 1997]** explica que idealmente duas seqüências S_1 e S_2 são definidas como sendo divergentes por 1 unidade PAM se uma série de mutação pontual aceita[57] (sem inserções ou exclusões) tiver convertido S_1 em S_2 (e vice-versa) com uma média de 1 ponto de mutação por 100 aminoácidos envolvidos. Isso não implica que, após 100 PAMs, cada aminoácido da seqüência será diferente: algumas posições podem mudar várias vezes, revertendo-se até mesmo aos aminoácidos originais, enquanto outras podem nem sofrer qualquer alteração.

As matrizes PAM e outras matrizes de substituição são geralmente apresentadas como matrizes de probabilidades logarítmicas (log-odds). Isso porque cada escore na matriz é o logaritmo de um “odds ratio”. O “odds ratio” usado é a razão do número de vezes que um resíduo “A” é observado em substituição ao resíduo “B”, dividido pelo número de vezes que se esperaria que um resíduo “A” substituisse o resíduo “B” de modo aleatório. Assim, um escore zero significa que a freqüência do par de aminoácidos no banco de dados é a mesma esperada *“por acaso”*; escores negativos designam pares de resíduos que se substituem menos freqüentemente do que se esperaria *“por acaso”* e evidenciam o fato de as seqüências não serem homólogas; escores positivos, por sua vez, designam um par de resíduos que se substituem mais freqüentemente do que se esperaria *“por acaso”*, e isso pode evidenciar homologia entre as seqüências **[PSC, 1999]**.

Por extrapolação, as demais matrizes da família PAM podem ser computadas: multiplica-se a matriz PAM1 por ela mesma *n* vezes e obtém-se a freqüência de mudanças para proteínas que tenham divergido *n*%, originando uma nova matriz PAM***n***.

Dessa forma, se a matriz PAM1 for multiplicada por ela mesma 250 vezes, obtém-se a matriz PAM250 (Tabela 3.3-12). Se duas seqüências divergirem 250%, espera-se que compartilhem aproximadamente 20% de identidade **[Pearson, 2001]**. Dado que 20% de identidade está no limite para se detectar uma similaridade significativa, **[Baxevanis e Ouellette, 2001]** as matrizes PAM200 e PAM250 têm sido largamente utilizadas para alinhamentos de seqüências bastante divergentes.

Desse modo, para alinhamentos de seqüências com um grau maior de similaridade, recomenda-se o uso das matrizes PAM de valores mais baixos, por exemplo PAM100, PAM120 e PAM160. Ou seja, uma matriz PAM, em particular, é mais eficiente para alinhar ou encontrar em um banco de dados seqüências que tenham divergido pela extensão indicada por sua unidade PAM.

A matriz PAM250

	C	S	T	P	A	G	N	D	E	Q	H	R	K	M	I	L	V	F	Y	W	
C	12																				**C**
S	0	2																			**S**
T	-2	1	3																		**T**
P	-3	1	0	6																	**P**
A	-2	1	1	1	2																**A**
G	-3	1	0	-1	1	5															**G**
N	-4	1	0	-1	0	0	2														**N**
D	-5	0	0	-1	0	1	2	4													**D**
E	-5	0	0	-1	0	0	1	3	4												**E**
Q	-5	-1	-1	0	0	-1	1	2	2	4											**Q**
H	-3	-1	-1	0	-1	-2	2	1	1	3	6										**H**
R	-4	0	-1	0	-2	-3	0	-1	-1	1	2	6									**R**
K	-5	0	0	-1	-1	-2	1	0	0	1	0	3	5								**K**
M	-5	-2	-1	-2	-1	-3	-2	-3	-2	-1	-2	0	0	6							**M**
I	-2	-1	0	-2	-1	-3	-2	-2	-2	-2	-2	-2	-2	2	5						**I**
L	-6	-3	-2	-3	-2	-4	-3	-4	-3	-2	-2	-3	-3	4	2	6					**L**
V	-2	-1	0	-1	0	-1	-2	-2	-2	-2	-2	-2	-2	2	4	2	4				**V**
F	-4	-3	-3	-5	-4	-5	-4	-6	-5	-5	-2	-4	-5	0	1	2	-1	9			**F**
Y	0	-3	-3	-5	-3	-5	-2	-4	-4	-4	0	-4	-4	-2	-1	-1	-2	7	10		**Y**
W	-8	-2	-5	-6	-6	-7	-4	-7	-7	-5	-3	2	-3	-4	-5	-2	-6	0	0	17	**W**
	C	**S**	**T**	**P**	**A**	**G**	**N**	**D**	**E**	**Q**	**H**	**R**	**K**	**M**	**I**	**L**	**V**	**F**	**Y**	**W**	

Tabela 3.3-12 – A matriz PAM250.

Os aminoácidos estão agrupados de acordo com as propriedades do Radical : C (sulfidril), STPAG (pequenos e hidrofílicos), NDEQ (ácido, amido ácido e hidrofílico), HRK (básicos), MILV (pequenos e hidrofóbicos) e FYW (aromáticos). Os valores estão expressos como logaritmos de base 10 (aproximadamente 1/3 dos valores bit). Baseado em **http://www.blc.arizona.edu/courses/bioinformatics/dayhoff.html**.

As matrizes da família PAM possuem vantagens teóricas sobre outros métodos alternativos de escore para alinhamentos. Do ponto de vista biológico, essas matrizes são construídas por meio de mutações observadas. Assim, elas contêm informações sobre o processo que gera mutações, bem como sobre os critérios importantes na seleção e no estabelecimento de mutação em uma certa população. Do ponto de vista estatístico, essas matrizes são as mais precisas descrições das mudanças dos aminoácidos, mudanças essas que são esperadas após um dado número de mutações e que podem ser derivadas de um banco de dados para criar as próprias matrizes **[PSC, 1999]**.

Dessa forma, o mais alto escore de alinhamento é estatisticamente o mais provável de ter sido gerado por evolução e não "por chance". O argumento estatístico aplica-se estritamente aos alinhamentos locais sem lacunas, mas talvez possa também ser utilmente aplicado àqueles com lacunas que são produzidos por muitos programas de busca em bancos de dados. Não está claro a que ponto esses argumentos estatísticos podem ser estendidos para os alinhamentos globais.

Um outro recente avanço é que o enfoque PAM tem sido aplicado para alinhamentos de ácidos nucléicos, o que levou a duas sugestões: a primeira é que se a seqüência é uma região codante, recomenda-se traduzi-la para uma seqüência de aminoácidos antes de se efetuar a busca em banco de dados; a segunda, e não menos importante, é que se for realmente necessário trabalhar com nucleotídeos, uma matriz PAM47 (47% de divergência) é a escolha apropriada de escores, pois isso corresponderia a um valor 5 para as identidades e –4 para as divergências. Sugestões para escores diferenciados de alinhamentos envolvendo transições e transversões são relatadas no artigo de States et al (1991)[xix].

Devido às estratégias utilizadas para se construir essa família de matrizes, algumas críticas têm sido apontadas **[BLC, 1999]** :

- Assume-se que cada posição ocupada por um aminoácido é igualmente mutável, embora, de fato, saiba-se que os sítios/domínios variem consideravelmente no seu grau de mutabilidade. Motivos altamente

[xix] D.J. **States**, W. Gish, and S.F. Altschul. *Improved sensitivity in nucleic acid database searches using application-specific scoring matrices*. Methods: A Companion to Methods in Enzymology, **1991**.

conservados de aminoácidos são encontrados em proteínas relacionadas. Os valores da matriz PAM baseiam-se apenas em dados dos domínios mais mutáveis. Além disso, a freqüência de alteração de aminoácido que requer duas alterações de nucleotídeos é maior que do que aquela que poderia ocorrer "*por acaso*".

- Embora alinhamento de proteínas relacionadas seja direto e bastante independente do esquema de escore por comparação de símbolos, alinhamentos de proteínas menos relacionadas é muito mais especulativo (teórico) e, contrariamente, depende do esquema[xx].
- E finalmente, as matrizes originais de Dayhoff são baseadas em um pequeno conjunto de proteínas relacionadas. Recentemente, esse conjunto foi atualizado para incluir um grupo maior de proteínas (vide [xxi] e [xxii]). Entretanto, não se encontrou informação adicional para aumentar significativamente a habilidade do sistema de escore para encontrar membros de mais famílias de proteínas[xxiii].

3.3.8 BLOSUM

Antes de se discutirem as matrizes BLOSUM propriamente ditas, serão tratados os dois bancos de dados envolvidos na sua construção: PROSITE e BLOCKS.

I. O banco de dados PROSITE

PROSITE é um banco de dados de famílias e domínios de proteínas e que está relacionado diretamente ao banco de dados de seqüências de proteínas SWISS-PROT. A sua construção, por Amos Bairoch, foi baseada na observação que, enquanto existe um número gigantesco de diferentes proteínas, a maioria delas pode ser agrupada pela similaridade de suas seqüências em um número limitado de famílias, pois, além da similiaridade, muitas vezes também compartilham atributos funcionais e são derivados de um ancestral comum **[Gusfield, 1997]**.

[xx] Feng D.F., Johnson M.S. and Doolittle R.F. (1985) Aligning amino acid sequences: comparison of commonly used methods.
[xxi] Gonnet, G.H., Cohen M.A., and Brenner S.A. (1992) Exhaustive matching of the entire protein sequence database.
[xxii] Jones D.T., Taylor W.R. and Thornton J.M. (1992) The rapid generation of mutation data matrices from protein sequences.
[xxiii] Henikoff S. and Henikoff J.G. (1992). Amino acid substitution matrices from protein blocks.

É aparente, ao se estudarem famílias de seqüências de proteínas, que algumas regiões permanecem mais bem conservadas que outras durante a evolução. Essas regiões são geralmente importantes para a função de uma proteína e para a manutenção de sua estrutura tridimensional. Ao se analisarem as propriedades constantes e variáveis de tais grupos de seqüências similares, é possível derivar um padrão biologicamente significativo ("uma impressão digital") para uma família ou um domínio de proteína, o qual distingue seus membros de outras proteínas não relacionadas. Esse padrão também pode ser usado para designar uma nova seqüência a uma família específica de proteínas e, assim, formular hipóteses sobre sua função **[PROSITE, 2002]**.

Um padrão PROSITE pode ser representado por uma assinatura ou por um perfil derivado de um alinhamento múltiplo dos membros de uma certa família. Atualmente este banco de dados contém assinaturas e perfis específicos para mais de mil famílias e domínios de proteínas. Cada um desses padrões vem acompanhado de documentação sobre a estrutura e a função da proteína da qual foi derivado **[Gusfield, 1997]**.

As assinaturas são escritas como expressões regulares[58] finitas, **[PROSITE, 2002]** utilizando as seguintes convenções:

- **a)** Uma letra designa cada aminoácido, segundo o padrão IUPAC (União Internacional de Química Pura e Aplicada).
- **b)** O símbolo "x" é usado para uma posição em que qualquer aminoácido é aceito.
- **c)** Ambigüidades são indicadas por uma lista entre colchetes dos aminoácidos aceitos para uma determinada posição. Por exemplo, [ALT] representa Alanina ou Leucina ou Treonina.
- **d)** Ambigüidades também são indicadas listando entre chaves os aminoácidos não aceitos para uma determinada posição. Por exemplo, {AM} representa qualquer outro aminoácido, exceto Alanina e Metionina.
- **e)** Cada elemento em um padrão é separado de seu vizinho por um "-".
- **f)** Repetição de um elemento do padrão pode ser indicada por um valor ou um limite numéricos colocados entre parênteses logo após esse elemento. Por exemplo, x(3) corresponde a x-x-x; e x(2,4) corresponde a x-x ou x-x-x ou x-x-x-x.

g) Quando o padrão está restrito ao N- ou C-terminal de uma seqüência, esse padrão tanto pode começar com "<" quanto terminar com ">", respectivamente. Em algumas situações raras (por exemplo, PS00267 ou PS00539), esse símbolo também pode ocorrer dentro dos colchetes para o elemento C-terminal, como é o caso de "F-[GSTV]-P-R-L-[G>].", que pode significar "F-[GSTV]-P-R-L-G." ou "F-[GSTV]-P-R-L>.".

h) Um ponto final encerra o padrão.

Alguns exemplos dessas assinaturas são:

a) "*[AC]-x-V-x(4)-{ED}.*", traduzido como *[Ala ou Cys]-qualquer-Val-qualquer-qualquer-qualquer-qualquer-{qualquer, exceto Glu ou Asp}*.

b) *"<A-x-[ST](2)-x(0,1)-V."*, que deve estar no N-terminal da seqüência ("<"), é traduzido para *Ala-qualquer-[Ser ou Thr]-[Ser ou Thr]-(qualquer ou nenhum)-Val*.

c) "*[RK]-x(2,3)-[DE]-x(2,3)-Y.",* o padrão PROSITE PS00007, traduz-se como *[Arg ou Lys]-qualquer-qualquer-[Asp ou Glu]-qualquer-qualquer-qualquer-Tyr*.

d) "*G-{EDRKHPFYW}-x(2)-[STAGCN]-{P}."*, o padrão PROSITE PS00008, que pode ser traduzido para *Gly-{qualquer, exceto Glu, Asp, Arg, Lys, His, Pro, The, Tyr ou Trp}-qualquer-qualquer-[Ser, Thr, Ala, Gly, Cys ou Asn]-{qualquer,exceto Pro}*.

Cada um dos padrões PROSITE representados como perfis são derivados de um alinhamento múltiplo de membros de uma família, e são usados quando a similaridade nesse conjunto é insuficiente para derivar uma assinatura efetiva. Habilidade adicional para reconhecer novos membros de uma família pode ser obtida por extrapolação do perfil, isto é, usando o conhecimento de substituição para dar escores para aminoácidos que ocorrem em certas posições nas quais não haviam sido observados. Por exemplo, se o aminoácido Leucina, e não a Isoleucina, é visto freqüentemente em uma dada posição do alinhamento múltiplo, a Isoleucina, mesmo assim, poderia receber um alto escore naquela posição porque esses dois aminoácidos possuem propriedades químicas similares e comumente se substituem **[Gusfield, 1997]**.

II. O banco de dados BLOCKS

BLOCKS é um banco de dados de motivos de proteínas, derivado inicialmente do PROSITE e desenvolvido por Steven e Jorja Henikoff em 1992 no Centro Fred Hutchinson de Pesquisa do Câncer (FHCRC), situado na cidade de Seattle em Washington, USA. Os motivos no BLOCKS são baseados na similaridade das subseqüências mais altamente conservadas em seqüências de proteínas distantemente relacionadas, mesmo que nenhuma função seja conhecida. Em contraste, os motivos armazenados no PROSITE foram coletados com atenção particular para funções e estruturas conhecidas de proteínas, ou seja, espera-se que cada um desses motivos tenha um significado biológico conhecido **[Gusfield, 1997]**.

Desde o ano de 1999, este banco de dados mudou sua estratégia e vem extraindo suas seqüências não mais de uma única fonte, **[InterPro, 2001]** mas de um consórcio de bancos de dados de proteínas chamado InterPro (Recurso Integrado de Famílias, Domínios e Sítios de Proteínas), do qual atualmente são membros Pfam[59], PRINTS[60], PROSITE, ProDom[61], SMART[62], TIGRFAMs[63] e SWISS-PROT + TrEMBL. Na versão 13.0, de agosto de 2001, este repositório já armazenava 8656 seqüências curtas e altamente conservadas, os chamados blocos[64], derivados de aproximadamente 2101 grupos de proteínas **[BLOCKS, 2001]**.

III. As matrizes BLOSUM

Este modelo foi desenvolvido por Steven e Jorja Henikoff em 1992, paralelamente ao seu trabalho de construção do banco de dados BLOCKS, descrito acima.

As matrizes BLOSUM possuem uma apresentação similar às matrizes PAM (uma matriz 20 x 20), mas seus desenvolvedores fizeram uso de uma estratégia diferente e de um conjunto muito maior de dados para estimar as freqüências-alvos. **[BLC, 1999]** Os valores das matrizes foram baseados na observação direta de substituição de aminoácidos de um conjunto de aproximadamente 2000 padrões conservados de aminoácidos, chamados blocos, que atuam como assinaturas das famílias de proteínas das quais derivam. **[Baxevanis e Ouellette, 2001]** Esses blocos foram encontrados no banco de dados BLOCKS, o qual contém alinhamentos múltiplos locais (blocos) envolvendo seqüências distantemente relacionadas, ao contrário do enfoque utilizado pela matriz

PAM. Embora não exista nenhum modelo evolucionário neste caso, é vantajoso ter dados gerados por observação direta e não por extrapolação.

Cada coluna do alinhamento dos blocos provia um conjunto de possíveis substituições de aminoácidos e consideraram-se as hipóteses: a) os eventos mutacionais são independentes do contexto; b) um acontecimento mutacional numa certa posição é independente dos eventos mutacionais anteriores que tiveram lugar nessa posição; c) a probabilidade de substituição de X por Y é a mesma que a de uma mudança reversa de Y resulte X; d) substituições mais comuns devem representar uma relação mais próxima entre dois aminoácidos de proteínas relacionadas e, por isso, recebem valores mais favoráveis no alinhamento; e e) contrariamente, substituições raras são menos favorecidas. Este procedimento, no entanto, poderia levar a uma representação excessiva das substituições que ocorrem na maioria dos membros de famílias de proteínas relacionadas **[BLC, 1999]**.

Para reduzir essa interferência dos membros mais relacionados, as seqüências dessas proteínas foram agrupadas em uma única seqüência antes de atribuir valores aos alinhamentos das seqüências dos blocos. Padrões com 62% de identidade foram novamente reagrupados para formar uma matriz de substituição chamada BLOSUM62 (Tabela 3.3-13), e aquelas com 80% de identidade formaram uma outra matriz chamada BLOSUM80, e assim por diante. Ou seja, **[Baxevanis e Ouellette, 2001]** do mesmo modo que acontece no modelo PAM, existe uma série numerada de matrizes BLOSUM, mas o número, neste caso, refere-se ao nível máximo de identidade que as seqüências possam ter e ainda contribuir independentemente para o modelo.

Desse modo, para comparar seqüências similares, foram construídas matrizes usando altos percentuais, enquanto baixos percentuais são mais apropriados para comparação de seqüências altamente divergentes **[Pearson, 2001]** De fato, as matrizes BLOSUM, em especial a BLOSUM62, atuam substancialmente melhor que as matrizes PAM com os procedimentos BLASTP e FASTA, mas ambas (BLOSUM e PAM) podem apresentar bons resultados quando usadas com penalidades ótimas para as lacunas.

As matrizes BLOSUM e PAM diferem não somente no modo pelo qual são construídas, mas também em seu uso. As matrizes PAM de baixos percentuais (PAM1,

PAM20, PAM40 etc) indicam o acontecimento de pouca mudança evolucionária. Já os altos números das matrizes BLOSUM (por exemplo, BLOSUM80), em contraste, é que indicam a mesma situação de pouca mudança evolucionária e alto grau de conservação das seqüências **[Pearson, 2001]**, conforme figura a seguir:

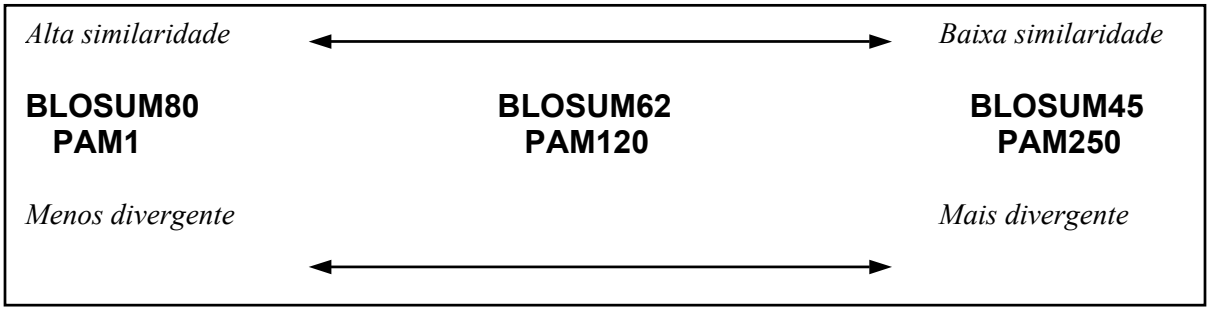

Figura 3.3-14 – Comparação do grau de divergência entre as matrizes BLOSUM e PAM.
Baseada em **http://www.ncbi.nih.gov/Education/BLASTinfo/Scoring2.html**.

A aprimoramento das matrizes BLOSUM em relação às matrizes PAM provavelmente deriva de dois fatores. O primeiro é que em 1992 - diferentemente das matrizes PAM construídas em 1978 - muitas outras seqüências de proteínas já haviam se tornado conhecidas e, assim, essas matrizes incorporaram novas observações de substituições de aminoácidos. O segundo fator é que as substituições usadas na construção das matrizes BLOSUM são aquelas substituições diretamente observadas em blocos bem conservados de alinhamento múltiplo das seqüências **[PSC, 1999],** e não por extrapolação de uma matriz inicial, o caso das matrizes PAM com relação à PAM1.

Limitando as substituições incluídas aos blocos bem conservados, isso leva a, pelo menos, dois benefícios: a) os alinhamentos são mais confiáveis nesses blocos e a proporção de substituições falsas ficam reduzidas; e b) igualmente importante, talvez esses blocos bem conservados sejam as regiões mais prováveis de serem encontradas nas buscas em bancos de dados e, assim, as matrizes BLOSUM representam os padrões de substituição mais apropriados **[PSC, 1999]**.

A matriz BLOSUM62																					
	C	S	T	P	A	G	N	D	E	Q	H	R	K	M	I	L	V	F	Y	W	
C	9																				C
S	-1	4																			S
T	-1	1	5																		T
P	-3	-1	-1	7																	P
A	0	1	0	-1	4																A
G	-3	0	-2	-2	0	6															G
N	-3	1	0	-2	-2	0	6														N
D	-3	0	-1	-1	-2	-1	1	6													D
E	-4	0	-1	-1	-1	-2	0	2	5												E
Q	-3	0	-1	-1	-1	-2	0	0	2	5											Q
H	-3	-1	-2	-2	-2	-2	1	-1	0	0	8										H
R	-3	-1	-1	-2	-1	-2	0	-2	0	1	0	5									R
K	-3	0	-1	-1	-1	-2	0	-1	1	1	-1	2	5								K
M	-1	-1	-1	-2	-1	-3	-2	-3	-2	0	-2	-1	-1	5							M
I	-1	-2	-1	-3	-1	-4	-3	-3	-3	-3	-3	-3	-3	1	4						I
L	-1	-2	-1	-3	-1	-4	-3	-4	-3	-2	-3	-2	-2	2	2	4					L
V	-1	-2	0	-2	0	-3	-3	-3	-2	-2	-3	-3	-2	1	3	1	4				V
F	-2	-2	-2	-4	-2	-3	-3	-3	-3	-3	-1	-3	-3	0	0	0	-1	6			F
Y	-2	-2	-2	-3	-2	-3	-2	-3	-2	-1	2	-2	-2	-1	-1	-1	-1	3	7		Y
W	-2	-3	-2	-4	-3	-2	-4	-4	-3	-2	-2	-3	-3	-1	-3	-2	-3	1	2	11	W
	C	S	T	P	A	G	N	D	E	Q	H	R	K	M	I	L	V	F	Y	W	

Tabela 3.3-13 – A matriz BLOSUM62.

Os aminoácidos estão agrupados de acordo com as propriedades do Radical : C (sulfidril), STPAG (pequenos e hidrofílicos), NDEQ (ácido, amido ácido e hidrofílico), HRK (básicos), MILV (pequenos e hidrofóbicos) e FYW (aromáticos). Comparada à matriz PAM160, esta matriz BLOSUM62 atribui valores positivos maiores para desigualdades entre aminoácidos raros (por exemplo, cisteína) e entre aminoácidos hidrofóbicos, e também valores mais negativos para desigualdades entre aminoácidos hidrofílicos. Baseado em **http://www.blc.arizona.edu/courses/bioinformatics/blosum.html**.

Na tabela acima (Tabela 3.3-13), **[BLC, 1999]** cada entrada é a freqüência atual de ocorrência do par de aminoácido no BLOCKS, agrupado com os demais de 62% de identidade, dividido pela probabilidade esperada de ocorrência. O valor esperado é calculado a partir da freqüência de ocorrência de cada um dos dois aminoácidos no BLOCKS, e provê uma medida de um alinhamento aleatório dos dois aminoácidos. A proporção atual/esperada é expressa como um escore "log–odds" das chamadas unidades *halfbit*, obtidas pela conversão da proporção para um logaritmo de base 2 multiplicado por 2. Um escore zero significa que a freqüência do par de aminoácidos no banco de dados é a mesma esperada *"por acaso";* um valor positivo mostra que o par foi encontrado mais freqüentemente que *"por acaso"*; e um escore negativo significa que o par foi encontrado menos freqüentemente que *"por acaso"*. Pode-se perceber que os mais

altos escores acontecem entre aminoácidos do mesmo grupo químico, algo também visto nas matrizes PAM.

3.3.9 Algoritmos heurísticos para buscas em bancos de dados

O advento de tecnologias rápidas e confiáveis para seqüenciamento de ácidos nucléicos e proteínas fez com que os atuais bancos de dados crescessem a uma proporção exponencial. Em uma aplicação típica de busca nessas bases, a operação básica é determinar quais das milhares de seqüências armazenadas apresentam maior similaridade com relação a uma particular seqüência de interesse. Os resultados são relatados como uma lista ordenada de ocorrências, seguida por uma série de alinhamentos individuais, vários escores e estatísticas.

Por isso, a escolha do programa de busca, do banco de dados e de vários outros parâmetros opcionais pode causar um impacto na efetividade dessa busca. Além do mais, existem várias interfaces para facilitar a interação, tais como comandos de estilo console, formulários Web e E-mails. Uma vantagem desse enfoque é que, para qualquer alinhamento observado, conseguem-se anotações completas e citações literárias que podem ser obtidas por meio das âncoras de hipertexto disponibilizadas **[Baxevanis e Ouellette, 2001]**.

Esse acúmulo de dados criou uma necessidade por programas eficientes para serem usados em buscas nesses repositórios. No entanto, a aplicação direta dos métodos de programação dinâmica tornou-se impraticável para essa tarefa por três motivos: a) a busca seria extremamente demorada; b) **[Baxevanis e Ouellette, 2001]** esses algoritmos concentraram-se no alinhamento de pares específicos, mas, para uma seqüência recém-determinada, não se teria nenhum caminho para saber qual a outra apropriada para usar em tal comparação; e c) **[Meidanis e Setubal, 1997]** a complexidade quadrática de tempo e espaço desses métodos fê-los inadequados para largas buscas em bancos de dados.

Para contornar esses problemas, uma solução seria empregar computadores paralelos e outros equipamentos especializados para continuar utilizando os já existentes algoritmos rigorosos. No entanto, o que está mais ao alcance são computadores de propósito geral e, por isso, é necessário recorrer a métodos baseados em heurísticas[65], os

quais fazem uso de aproximações para significativamente acelerar as comparações de seqüências, embora não garantam que as melhores identidades sejam encontradas e ainda apresentem um pequeno risco de que alinhamentos de altos escores sejam perdidos **[Baxevanis e Ouellette, 2001]**. Em geral, esses métodos dificultam o estabelecimento da complexidade teórica de espaço e tempo, mas, apesar disso, têm-se mostrado ferramentas muito importantes **[Meidanis e Setubal, 1997]**.

Um método heurístico é baseado na estratégia de particionamento de uma seqüência em pedaços menores de caracteres consecutivos, as chamadas *palavras*. Métodos baseados em palavras foram introduzidos no início da década de 1980 e são usados, na prática, por todos os programas populares de busca atuais. A idéia básica é que um alinhamento representando um relacionamento verdadeiro entre duas seqüências conterá, pelo menos, uma palavra que seja comum a ambas. As ocorrências podem ser identificadas de modo muito rápido pela pré-indexação de todas as palavras da seqüência de busca e posterior consulta a esse índice à medida que o banco de dados vai sendo rastreado **[Baxevanis e Ouellette, 2001]**.

Dois algoritmos heurísticos rápidos são freqüentemente usados para busca em bancos de dados de seqüências de DNA e proteína: BLAST (*Basic Local Alignment Search Tool* ou Ferramenta Básica de Busca de Alinhamento Local) e FASTA (*Fast Alignment* ou Alinhamento Rápido). **[Meidanis e Setubal, 1997]** Nenhum deles usa programação dinâmica pura, embora FASTA execute uma variante do método de Smith-Waterman para refinar alinhamentos obtidos por outros de seus procedimentos. **[Pearson, 2001]** Esses métodos chegam a ser 5 a 50 vezes mais rápidos que os algoritmos rigorosos de programação dinâmica, e podem produzir resultados de qualidade similar em muitos casos porque examinam apenas uma porção dos alinhamentos potenciais entre duas seqüências.

I. FASTA

Este algoritmo é um enfoque heurístico para alinhamentos rápidos de pares de seqüências de DNA e proteína e que tenta aproximar o melhor alinhamento local e o escore, reduzindo o custo computacional que ocorreria pela aplicação direta do procedimento Smith-Waterman. **[Baxevanis e Ouellette, 2001]** A primeira versão do FAST, o

FASTP, foi desenvolvida por William R. Pearson and David J. Lipman em 1985. **[Meidanis e Setubal, 1997]** Em 1988, após alguns aprimoramentos no algoritmo original, surgiu o FASTA. A seguir (Tabela 3.3-14), uma lista completa dos algoritmos dessa família **[Pearson, 2002]**:

Algoritmo	Funcionalidade
FASTA	Compara uma seqüência de proteína a outra seqüência ou biblioteca de proteína, ou compara uma seqüência de DNA a outra seqüência ou biblioteca de DNA.
TFASTA	Compara uma seqüência de proteína a uma seqüência ou biblioteca de DNA, traduzindo cada seqüência de DNA em todos as 6 possíveis janelas (*frames)* de leitura (3 para frente e 3 janelas reversas sem alterações na janela de leitura (*frameshifts*[66])) e comparando cada janela a uma seqüência de proteína**.**
FASTX / FASTY	Compara uma seqüência de DNA a uma seqüência de proteína do banco de dados. Traduz a seqüência de DNA em 3 janelas com alterações na janela de leitura. **FASTY2** permite alterações na janela de leitura dentro dos códons.
TFASTX / TFASTY	Compara uma seqüência de proteína a uma seqüência de DNA do banco de dados traduzida, com alterações na janela de leitura. **TFASTY** permite alterações na janela de leitura dentro dos códons.
FASTF / TFASTF	Compara uma mistura ordenada de peptídeos (obtida, por exemplo, da degradação Edman da segmentação CNBr) a um banco de dados de proteína ou DNA traduzido.
FASTS / TFASTS	Compara uma série de pequenos fragmentos de peptídeos (obtida de uma análise de espectometria de massa de uma proteína) a um banco de dados de proteína ou DNA traduzido.

Tabela 3.3-14 – Os algoritmos da família FAST.

Mais que comparar resíduos individuais em duas seqüências, FASTA procura por identidade de padrões ou palavras (*k-tuplas)* e, então, tenta construir um alinhamento local baseado nessas identidades. Devido às suas alta velocidade e sensitividade, tornou-se bastante útil para especificar a função de seqüências recém-determinadas em laboratório, o que é feito por meio de comparações com outras seqüências relacionadas, disponíveis nos atuais banco de dados. FASTA é comparável ao BLAST no que diz respeito ao algoritmo e à confiabilidade, mas pode ser mais sensível a buscas de famílias

de proteínas. Recomenda-se seu uso também para buscas de seqüências de DNA, ao invés do uso BLAST **[BLC, 1999]**.

São 6 as etapas executadas pelo algoritmo:

1. ***Localizar os "hot spots"*** :

   FASTA permite a especificação de um parâmetro chamado k-tuplas (abreviação de k respectivas tuplas), que determina o tamanho base da palavra para comparação entre a seqüência de busca e uma outra no banco de dados. Os valores k-tuplas são tipicamente 4 ou 6 para seqüências de DNA; e, para seqüências de proteína, FASTA focaliza regiões onde existam identidades aos pares (k-tuplas=*2*) ou unitárias (k-tuplas=1) **[Gusfield, 1997]**. As subseqüências idênticas com tamanho k-tuplas são referenciadas como "hot spots". **[Pearson, 2001]** Se a opção -*S* for usada, blocos de baixa complexidade serão ignoradas nas etapas a seguir.

   O valor k-tuplas afeta a performance do alinhamento no que concerne à sua sensitividade e seletividade[67]. Quanto ao FASTA e outros programas da mesma família, baixos valores k-tuplas aumentam a sensitividade e são desfavoráveis para a seletividade, enquanto altos valores diminuem a sensitividade e aumentam a seletividade, ou seja, esses dois conceitos agem de modo antagônico **[Meidanis e Setubal, 1997]**.

   Sejam *s* e *t* duas seqüências de proteína a serem comparadas, cujos tamanhos são denotados por $m = |s|$ e $n = |t|$. A comparação se inicia pela determinação das k-tuplas (= 1 ou 2) comuns a ambas seqüências. Em adição, o *offset* (contrabalanço) de uma k-tupla comum é importante no algoritmo. Esse *offset* é um valor entre $-n+1$ e $m-1$, que determina o deslocamento de uma seqüência em relação à outra, ou seja, se uma k-tupla comum começa na posição $s[i]$ e $t[j]$, diz-se que o offset é $(i - j)$.

   As seguintes estruturas de dados são necessárias : a) uma tabela de busca (*lookup table*) e b) um vetor indexado pelos *offsets* com valor inicial igual a zero. A seqüência *s* é rastreada e a tabela de busca,contendo todas as posições de uma certa k-tupla em *s*, vai sendo construída. **[Gusfield, 1997]**

Quanto a DNA, isso significa 4^6 (4 possibilidades de letras do alfabeto ACGT para cada posição de uma palavra de tamanho k-tuplas=6) ou 4096 possíveis entradas na tabela. Quanto a aminoácidos, têm-se 20^2 (20 possibilidades de letras do alfabeto de aminoácidos para cada posição de uma palavra de tamanho k-tuplas=2) ou 400 entradas na tabela. Para o caso de DNA, cada palavra é representada como um número de base 4, que também é o índice da tabela. Cada entrada contém os *offsets* onde essa particular combinação de 6 letras ocorre na seqüência de busca. Desse modo, para cada palavra na seqüência-alvo, apenas o dicionário precisa ser consultado para determinar se e onde a palavra ocorre na seqüência de busca.

Logo em seguida, a seqüência *t* é também examinada, e cada uma de suas k-tuplas é confrontada com as entradas da tabela de busca. Para todas as ocorrências comuns, o *offset* correspondente no vetor é incrementado. A Figura 3.3-15 exibe o conteúdo final da tabela e do vetor para as seqüências s=HARFYAAQIVL e t=VDMAAQIA. É possível notar que o offset +2 tem o mais alto valor de entrada, significando que muitas identidades foram encontradas para ele [**Meidanis e Setubal, 1997**].

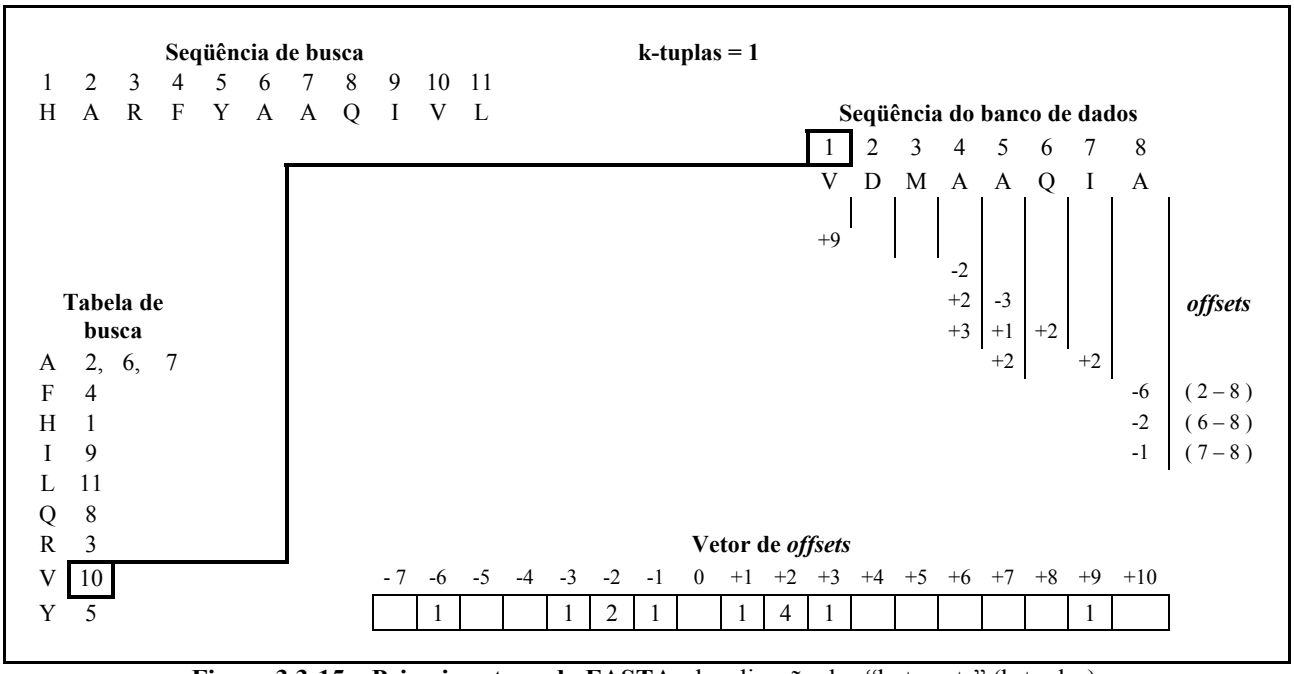
Seqüência de busca
k-tuplas = 1
1 2 3 4 5 6 7 8 9 10 11
H A R F Y A A Q I V L
Seqüência do banco de dados
1 2 3 4 5 6 7 8
V D M A A Q I A
+9
-2
+2 -3
+3 +1 +2
+2 +2
-6 (2 – 8)
-2 (6 – 8)
-1 (7 – 8)
offsets
Tabela de busca
A 2, 6, 7
F 4
H 1
I 9
L 11
Q 8
R 3
V 10
Y 5
Vetor de offsets
- 7 -6 -5 -4 -3 -2 -1 0 +1 +2 +3 +4 +5 +6 +7 +8 +9 +10
1 1 2 1 1 4 1 1

Figura 3.3-15 – Primeira etapa do FASTA : localização dos "hot spots" (k-tuplas).
Baseado em [**Meidanis e Setubal, 1997**].

2. ***Encontrar as 10 melhores regiões utilizando uma matriz de substituição:***

Num procedimento mais detalhado, duas ou mais k-tuplas comuns são combinadas quando estão próximas e na mesma diagonal. Os critérios são heurísticos. Essas k-tuplas ("hot spots") combinadas são chamadas de região. Esse método é conhecido como o método da diagonal, pois um *offset* pode ser visto como uma diagonal em uma matriz de programação dinâmica (Figura 3.3-16). Um possível uso do mais alto offset é executar o algoritmo para outras diagonais próximas. Uma região pode ser considerada como um par de segmentos (na terminologia BLAST) ou como um alinhamento local sem lacunas. Às regiões são atribuídos escores que reflitam suas identidades e desigualdades **[Meidanis e Setubal, 1997]**. As palavras que se sobrepõem na mesma diagonal não são consideradas no escore. É importante lembrar-se de que regiões não contêm lacunas e nem *indel* (inclusão ou exclusão) porque ela é derivada de uma única diagonal **[Gusfield, 1997]**.

Após isso, rastream-se as 10 regiões com a mais alta densidade de identidades usando uma matriz de substituição, que pode ser baseada em aminoácidos ou nucleotídeos, por exemplo a BLOSUM50. Para incluir somente aqueles resíduos que contribuem para o escore mais alto, truncam-se os finais da região **[Pearson, 2001]**.

Esses melhores 10 escores são a primeira medida da similaridade entre *s* e *t.* O único e melhor subalinhamento encontrado após a aplicação da matriz de escore é chamado de *init1*, que é computado para cada comparação entre a seqüência de busca e uma seqüência do banco de dados. Os valores são relatados num histograma, juntamente com o escore médio. O *init1* é também usado para ordenar todas as seqüências do banco de dados **[Meidanis e Setubal, 1997]**.

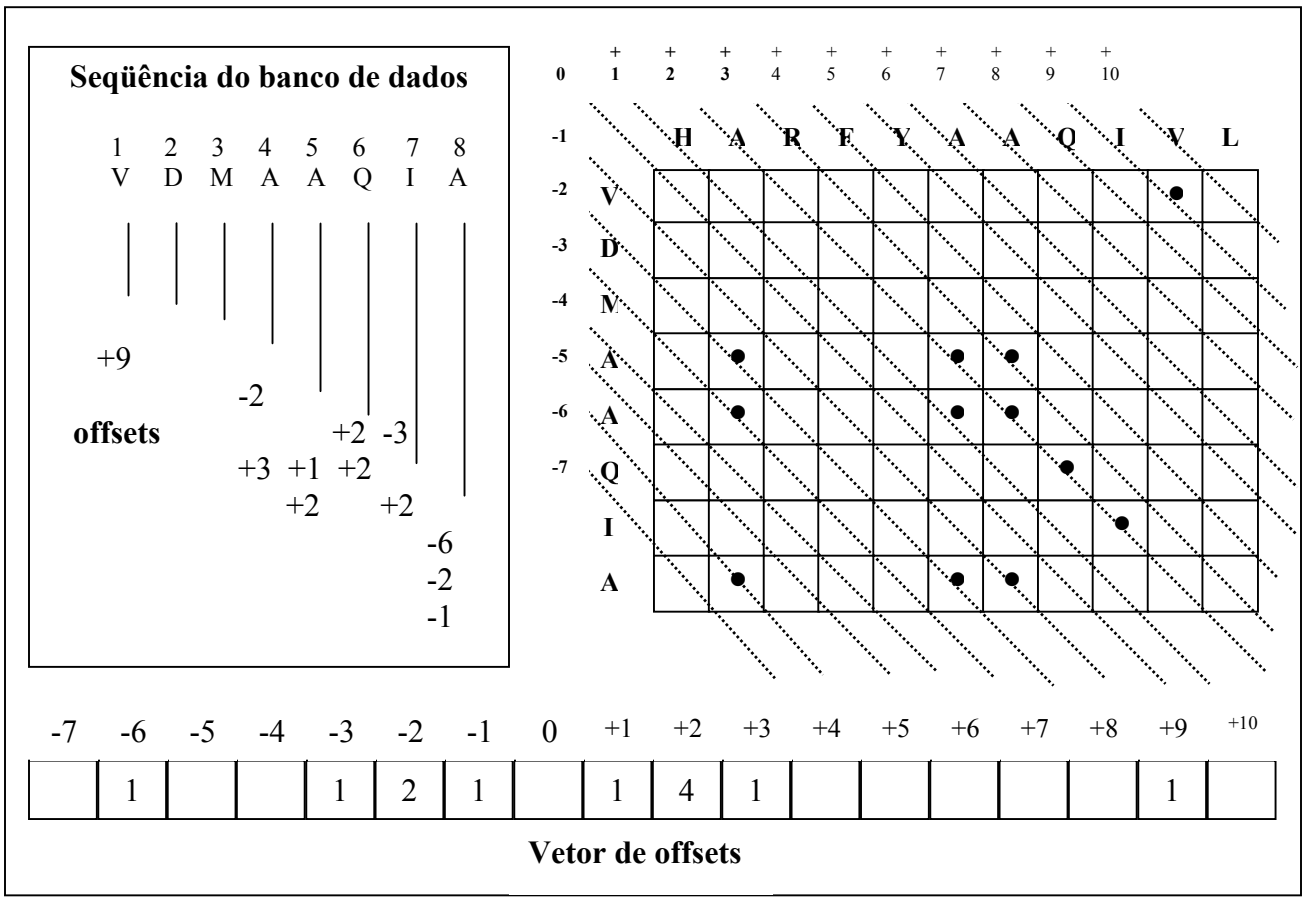

Figura 3.3-16 – Segunda etapa do FASTA : método da diagonal.
Baseado em [**Meidanis e Setubal, 1997**].

3. ***Combinar regiões iniciais de diagonais diferentes***:

Se existirem várias regiões iniciais com escores maiores que o valor *CUTOFF* (atalho), o algoritmo checa se as regiões iniciais truncadas podem ser combinadas para formar um alinhamento aproximado sem lacunas (uma nova região de alto escore) **[Pearson, 2001]**. Esse escore é a soma dos escores das regiões contribuintes, subtraída da penalidade (geralmente 20) de cada junção. Regiões que se sobrepõe não podem ser combinadas. Para que as junções possam ser feitas, inserções ou exclusões são introduzidas. O mais alto escore para uma região, após esta etapa, é chamado de *initn* **[Dipperstein, 2000]**. Vide Figura 3.3-17 a seguir.

Nesta etapa, FASTA tenta combinar bons subalinhamentos em um único alinhamento de mais alto escore e que permite alguns espaços. A estratégia geral pode ser explicada de outro modo: cada um dos 10 melhores

subalinhamentos da última etapa (escore acima do valor *CUTOFF* (atalho)) é representado por um nó valorado em um grafo direcionado, sendo esse valor igual ao seu escore **[Gusfield, 1997]**.

Seja *u* a representação de um dos subalinhamentos, começando na posição (*i, j*) e terminando em (*i + d, j + d*) na tabela; e seja *v* um outro dos subalinhamentos começando na posição (*i', j'*). Então, estende-se uma reta do nó *u* até o nó *v* no grafo se, e somente se, *i' > i + d*. Ou seja, *v* deve começar em uma linha antes do término de *u* na tabela. Aplica-se um valor a cada reta para penalizar qualquer lacuna que seja criada no alinhamento, onde o subalinhamento *u* é seguido pelo *v*. Uma penalidade maior (negativa) deve ser aplicada ao vértice (*u, v*) se *i'* for maior que *i + d*, ou se existir uma distância maior entre as duas diagonais que contêm *u* e *v*. Alternativamente, pode-se usar uma penalidade constante para uma lacuna, independente de quantos espaços a componha. Essencialmente, FASTA tentará encontrar um único caminho de valor máximo do grafo, que gerará uma saída denominada *initn*.

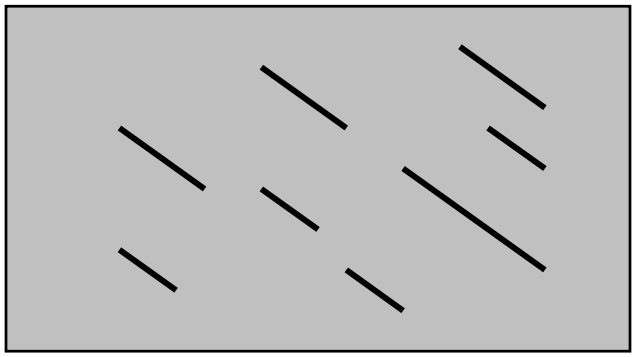

Figura 3.3-17 – Terceira etapa do FASTA : junção das regiões iniciais de diagonais diferentes. Baseada em [Martins, 2000].

4. ***Determinar o alinhamento ótimo***:

Para o escore mais elevado das 10 seqüências anteriores, um escore otimizado (*opt*) é computado pela execução do algoritmo de programação dinâmica restrito a uma faixa em torno do alinhamento inicial (aquele que produziu o *init1*) **[Meidanis e Setubal, 1997]**. Para buscas envolvendo proteínas

com k-tuplas=2, uma faixa de 16 resíduos é usada por definição; se k-tuplas for igual a 1, essa faixa será de 32 resíduos **[Pearson, 2001]**.

Na prática, quando seqüências são verdadeiramente relacionadas, o escore otimizado costuma ser muito mais significativo que o escore inicial. Essa observação freqüentemente ajuda a distingüir entre bons alinhamentos ocorrendo *"por acaso"* e relacionamentos verdadeiros **[Meidanis e Setubal, 1997]**. Vide Figura 3.3-18 a seguir:

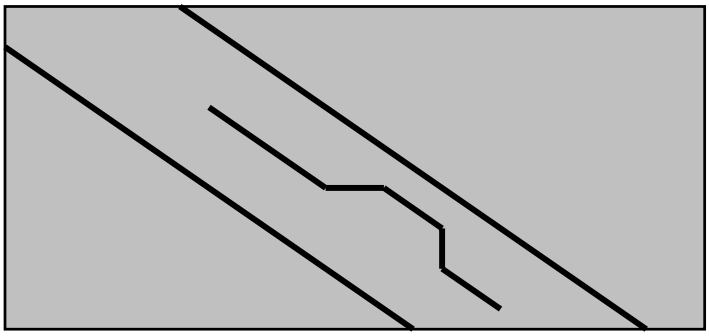

Figura 3.3-18 – Quarta etapa do FASTA : determinação do alinhamento ótimo. Baseada em [Martins, 2000].

5. ***Calcular as estimativas estatísticas***:

Após o cálculo dos 60.000 primeiros escores, normalizam-se os escores brutos de similaridade usando estimativas para os parâmetros estatísticos de distribuição extrema de valor. A estratégia padrão determina o escore da similaridade contra o tamanho da biblioteca de seqüências e calcula a variência média. Os valores Z (escores normalizados com média 0 e variância 1) são calculados, e o cálculo é repetido para seqüências do banco de dados com valores Z maiores que 5.0 e menores que –5.0 removidos **[Pearson, 2001]**.

6. ***Apresentar o resultado***:

Finalmente o banco de dados é ordenado pelos escores *opt* ou *initn* e as seqüências de resultados mais altos são submetidas a um alinhamento pelo algoritmo Smith-Waterman (sem limitação do tamanho de lacuna). O número de alinhamentos feitos deste modo depende do escopo da busca definido pelo usuário.

II. BLAST

Quase imediatamente ao seu lançamento em 1990, BLAST tornou-se a ferramenta dominante para busca de seqüências em bancos de dados. As razões iniciais para esse sucesso deveram-se à velocidade, ao fato de que é apresentada uma saída com várias soluções e a que cada identidade é acompanhada de uma estimativa de significância estatística (essencialmente a probabilidade de que uma identidade de um certo valor ou melhor possa ser encontrada em seqüências aleatórias). Por isso, desde a introdução do BLAST, FASTA tem evoluído e agora também apresenta uma saída de dados com as mesmas características. Mesmo assim, BLAST ainda continua dominante **[Gusfield, 1997]** .

Pode-se afirmar que BLAST originou-se da convergência de três objetivos. O primeiro foi um esforço geral de David J. Lipman, Warren Gish e outros de seus colegas no NCBI para aumentar a velocidade do FASTA pela introdução de regras mais severas para localizar (menos e melhores) alinhamentos dos "hot spots". O segundo foi um trabalho de busca sublinear de Myers[xxiv], o qual introduziu as idéias de vizinhança de subséries e máquina de estados finitos para localizar os "hot spots" iniciais. O terceiro foi o trabalho de Karlin, Altschul e Dembo, que derivaram os resultados de probabilidade usados no BLAST para avaliar a significância estatística das identidades relatadas.

BLAST é atualmente uma coleção de programas (Tabela 3.3-15), sendo que cada um deles está direcionado para um domínio diferente **[NCBI/Education, 2002]**:

Programa	Funcionalidade	*Frames*
blastp	Compara duas seqüências de aminoácidos, sendo a primeira a seqüência de busca e a segunda, do banco de dados.	1
blastn	Compara duas seqüências de nucleotídeos, sendo a primeira a seqüência de busca e a segunda, do banco de dados.	1
blastx	Compara uma seqüência de nucleotídeo (traduzida em todos os seus *reading frames*) com uma seqüência de proteína do banco de dados. Pode-se usar esta opção para encontrar potenciais produtos traducionais (*potential translation products*) de uma seqüência desconhecida de	6

[xxiv] Vide E. W. **Myers**. A **sublinear** algorithm for approximate keyword searching. Algorithmica, 12(4/5):345-374, 1994.

	nucleotídeo.	
tblastn	Compara uma seqüência de proteína com uma seqüência de nucleotídeo traduzida dinamicamente em todos os seus *reading frames*.	6
tblastx	Compara os 6 *frames* traducionais de uma seqüência de nucleotídeos com os 6 *frames* traducionais de outra seqüência de nucleotídeos do banco de dados. Não se permite que esta opção seja usada com o banco de dados *nr* diretamente na página do BLAST, pois ela demanda recursos computacionais muito intensos.	36

Tabela 3.3-15 – Os algoritmos da família BLAST.

Além da seleção do programa de BLAST, deve-se também preocupar com a escolha do tipo de banco de dados NCBI a ser utilizado para comparar com a seqüência de busca. Alguns desses bancos são específicos para proteínas ou nucleotídeos e não podem ser usados em combinação com certos programas, por exemplo uma busca BLASTN com SWISSPROT **[NCBI/Education, 2002]**. Quanto aos bancos de dados de proteínas, esses estão agrupados por não-redundância, periodicidade de atualização, tipos de organismos ou espécies, seqüências patenteadas, interesse imunológico, elementos repetitivos etc, agregando dados do GenBank, PDB, SWISSPROT, PIR, PRF(Fundação Japonesa de Pesquisa de Proteína), REPBASE[68] e BDGP (Projeto Berkeley de Genoma de Drosophila). Quanto aos bancos de dados de nucleotídeos, além de agrupamentos similares aos relatados para proteínas, existem outros de EST[69], STS[70], vetores, seqüências de mitocôndrias, GSS[71], seqüências HTGS[72] etc, agregando dados do GenBank, EMBL, DDBJ, PDB, REPBASE e BDGP. Informações mais aprofundadas podem ser obtidas no Tutorial de Busca do BLAST (http://www.ncbi.nlm.nih.gov/Education/BLASTinfo/query_tutorial.html) ou na lista de seus bancos de dados disponíveis (http://www.ncbi.nlm.nih.gov/BLAST/blast_databases.html).

BLAST retorna uma lista dos pares de segmentos com alta similaridade local em alinhamentos sem lacunas, avaliados por uma matriz de escore, quando da comparação entre a seqüência de busca e as seqüências do banco de dados. Alinhamentos com algumas lacunas podem ser criados pelo encadeamento de várias dessas regiões. No entanto, antes de se estudar a lógica deste algoritmo, é necessário introduzir a terminlogia

básica quanto a segmentos e, com isso, manter a consistência da discussão quanto à publicação original do BLAST **[Meidanis e Setubal, 1997]** :

- **Segmento**: é uma subsérie de uma seqüência.
- **Par de segmentos**: dadas duas seqüências *s e t*, um par de segmentos entre elas são duas subséries de mesmo tamanho, cada uma derivada de uma dessas seqüências envolvidas. Devido ao fato de possuirem o mesmo tamanho, pode-se formar um alinhamento sem lacunas (e sem função de penalidade), ao qual se pode atribuir um escore utilizando uma matriz de substituição. Segue abaixo um exemplo no qual se determina o escore pela matriz BLOSUM62:

s	**...**	**..**	**...**	**R**	**L**	**A**	**M**	**K**	**...**	**...**	**...**	
t	**...**	**...**	**...**	**S**	**K**	**A**	**N**	**V**	**...**	**...**	**...**	
Escore	**...**	**...**	**...**	**-1**	**-2**	**+4**	**-2**	**-2**	**...**	**...**	**...**	**= -3**

Tabela 3.3-16 – Um par de segmentos com escore determinado pela matriz BLOSUM62.

- **Par de segmentos máximo local**: é um par de segmentos cujo escore do alinhamento poderia diminuir pela expansão ou redução desses segmentos em qualquer das direções (esquerda ou direita) **[Gusfield, 1997] .**
- **Par de segmentos máximo**: ou segmento máximo, é um par de segmentos cujo escore do alinhamento é o maior dentre todos aqueles determinados entre as duas seqüências envolvidas. Esse escore é uma medida de similaridade e pode ser computado por programação dinâmica, mas o BLAST consegue determiná-lo bem mais rápido.

Ao comparar todas as seqüências em um banco de dados com uma seqüência de busca P, BLAST tenta encontrar as seqüências que contenham um par de segmentos máximo com escore acima de um valor C (escore do segmento), segmentos também chamados de “sementes” (curtos pares de segmentos). A escolha desse valor C é guiada por teoremas, os quais se baseiam na matriz de substituição, nas características de P e nas seqüências do banco de dados. Esses teoremas identificam o mais baixo valor de C para o qual um par de segmentos máximo com escore C não ocorreria *“por acaso”* em qualquer das seqüências do banco de dados **[Gusfield, 1997]** . As sementes são estendidas

tanto para a esquerda quanto para a direita, sem inclusão de lacunas, até que o máximo escore possível seja atingido **[Meidanis e Setubal, 1997]**.

É verdade que nem todas as extensões são consideradas, principalmente porque o programa tem um critério de parada quando esses escores ficam abaixo de um determinado limite. Existe uma pequena chance de que as extensões corretas não sejam encontradas devido a essa otimização de tempo, mas, na prática, isso é altamente aceitável. **[Gusfield, 1997]** Portanto, qualquer seqüência com um par de segmentos máximo de escore acima de C é considerada "significante" e, então, relatada. BLAST também relata seqüências que não apresentam essa característica mas possuem vários pares de segmentos que, em combinação, mostrem-se estatisticamente significantes.

Refinamentos ao programa original foram implementados para aumentar a velocidade (necessária devido ao crescimento dos bancos de dados), permitir alinhamentos com lacunas e para aumentar a sensitividade das buscas de proteínas usando perfis gerados (PSI-BLAST). Durante a análise, duas seqüências são comparadas utilizando o enfoque abaixo, descrito em 4 passos. Os passos 1 e 2 são do programa original. A versão atual do BLASTP usa a observação de que, ao invés de estender cada palavra como é feita no passo 2, é melhor usar palavras mais curtas e isso exige que as duas palavras coincidentes estejam próximas na mesma diagonal antes de se computar um par de segmentos máximo **[Bioinfo_326, 2002]**.

Devido ao BLAST com lacuna (*Gapped BLAST*), foi adicionado o passo 3, no qual é utilizado um valor maior para o escore máximo para que a execução seja do mesmo nível que aquela obtida com o BLAST sem lacuna (*Ungapped BLAST*). A razão é que se um alinhamento interessante contém, por exemplo, 3 pares de segmentos máximo críticos, então o BLAST sem lacuna precisa de um limite maior que o mínimo dos escores desses segmentos, mas o BLAST com lacunas pode usar o valor máximo desses escores, pois, uma vez encontrado um desses segmentos máximos, a fase de extensão sem lacuna encontrará as demais. Isso é 3 vezes mais rápido que a execução do programa original. O passo 4, por sua vez, é usado pelo PSI-BLAST.

Seguem os 4 passos para execução do procedimento:

1. ***Compilar uma lista com palavras de alto escore, sendo que cada uma significa uma "semente"***:

As palavras coincidentes (*word hits*) são identificadas. Por palavra entenda-se um segmento contíguo (subsérie) de uma das seqüências e que tem um tamanho pré-determinado (tal como 10 para seqüência de DNA e 3 ou 4 para seqüência de proteína). Desse modo, duas palavras coincidentes são um par de segmentos muito similares, derivadas das seqüências envolvidas. Para o caso de DNA, as seqüências devem ser idênticas (baseando-se numa matriz de escore unitária[73]), enquanto que para proteínas, as seqüências devem apresentar alto escore de alinhamento (baseado em alguma das matrizes de substituição, por exemplo PAM ou BLOSUM). Este processo é extremamente rápido, consumindo tempo proporcional à soma dos tamanhos das seqüências **[Bioinfo_326, 2002]**.

Vide Figura 3.3-19 a seguir:

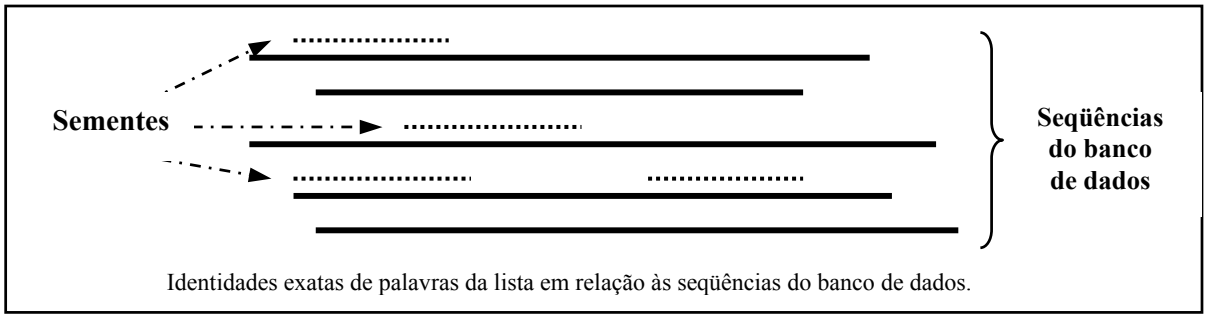
Sementes
Seqüências do banco de dados
Identidades exatas de palavras da lista em relação às seqüências do banco de dados.

Figura 3.3-19 – BLAST: *Compilação de uma lista com palavras de alto escore ("semente").* Baseado em [Martins, 2000].

Para seqüências de proteína, a lista de palavras de alto escore consiste de todas as palavras com W caracteres (chamados de w-mers) que apresentem um escore maior ou igual a T (escore da palavra) quando comparadas à seqüência de busca, usando uma matriz de substituição para computar os escores. W e T são parâmetros do programa, além de S (escore mínimo de segmento). Essa lista talvez não contenha todas as identidades de tamanho W. Se essas identidades consistem de aminoácidos muito comuns, as seqüências podem ser descartadas porque mesmo um alinhamento da seqüência com sua cópia poderia apresentar um valor

abaixo de T. Entretanto, existe uma opção para forçar a inclusão de todas essas seqüências de baixa complexidade **[Meidanis e Setubal, 1997]**.

Para buscas de DNA, a lista inicial contém apenas as palavras de tamanho W da seqüência de busca. Determinar escore para essas seqüências é sempre mais fácil e isso é suficiente para todos os propósitos. A estratégia de rastreamento é radicalmente diferente do caso das proteínas. Há a vantagem do fato de que o alfabeto tem apenas um tamanho 4 (ACGT) e, por isso, o banco de dados é inicialmente comprimido para permitir que cada nucleotídeo seja representado por 2 bits, ou seja, 1 byte comporta 4 nucleotídeos. Além do espaço que se economiza, a busca torna-se mais rápida porque 1 byte é comparado a cada vez (Tabela 3.3-17). Existe um passo extra de filtragem que remove dessa lista inicial palavras muito comuns do banco de dados, o que evita um alto número de falsas identidades.

Seqüência de busca : **QLNFSAGWAA**

Tamanho da palavra W = 2 (tipicamente esse valor é 3 ou 4 para seqüências de proteínas)
Escore da palavra T = 8

Palavras de W=2 na seqüência de busca	**Escore das palavras do banco de dados quando alinhadas às palavras da seqüência de busca**				
QL	**QL = (5+4),**	~~QM = (5+2),~~	~~YL = (1+4)~~	~~RM = (1+2)~~	...
LN	**LN = (4+6),**	~~LD = (1+1),~~	~~LM = (1 2)~~	~~ID = (2+1)~~	…
NF	**NF = (6+6),**	~~AF = (2+6),~~	**NY = (6+3),**	~~DF = (1+6)~~	…
FS	**FS = (6+4),**	~~FA = (6+1),~~	~~FV = (6 2),~~	~~MT = (0+1)~~	...
SA	**SA = (4+4),**	~~SD = (4 2),~~	~~SW = (4 3),~~	~~AW = (1 3)~~	…
AG	**AG = (4+6),**	~~AC = (4 3),~~	~~WG = (3+6),~~	~~NW = (2 2)~~	…
GW	**GW =(6+11),**	**GY = (6+2),**	**AW = (0+11),**	**WW = (-2+11)**	...
WA	**WA =(11+4),**	…	...	...	...
AA	**AA =(4+4),**	…	...	...	...

Tabela 3.3-17 – BLAST: Seqüência de busca, parâmetros e palavras com alto escore (matriz BLOSUM62).
Baseado em [**Craven, 2002**].

Há dois enfoques usados para rastrear o banco de dados em busca de palavras na lista anteriormente construída. Um deles é construir uma tabela de busca (*hash table* ou *lookup table*) de todas as palavras na primeira seqüência e, então, comparar cada palavra da segunda seqüência

com as entradas da tabela **[Meidanis e Setubal, 1997]**, como se pode observar na Figura 3.3-20 a seguir:

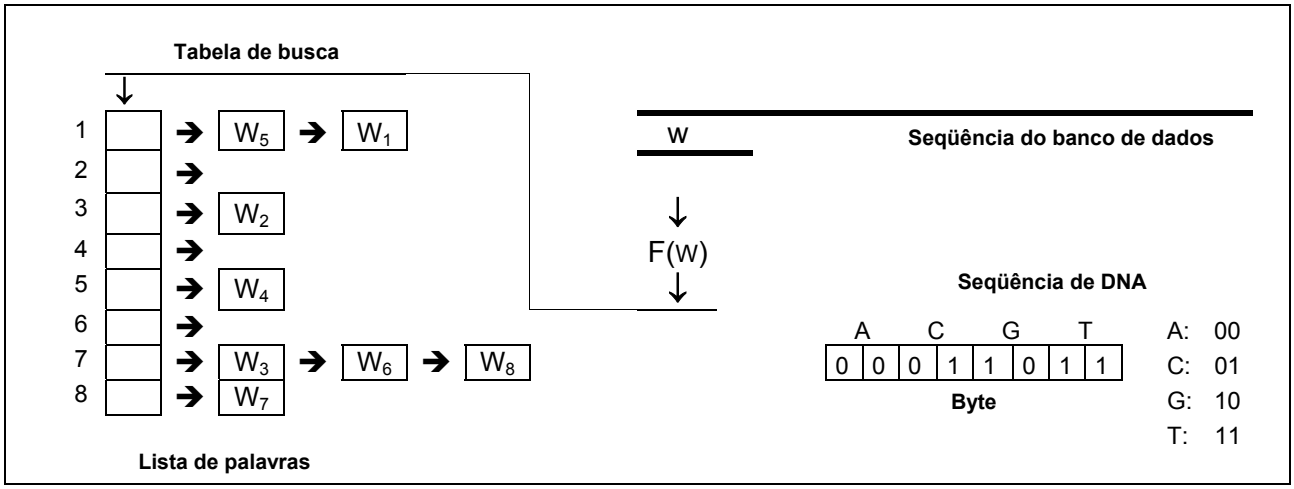

Figura 3.3-20 – BLAST: *Tabela de busca e comparação das seqüências do BD com as entradas desta tabela***.** Baseada em [Martins, 2000].

O segundo enfoque usa um autômato finito determinístico (*DFA - Deterministic Finite Automaton*) para buscar as palavras, conforme Figura 3.3-21. Esse dispositivo possui estados e transições e opera como uma máquina. Ele inicia em um estado inicial fixo, e para cada caracter no banco de dados, uma transição é feita para um outro estado. Dependendo do estado e da transição, uma palavra da lista é reconhecida. O autômato é construído somente uma vez, usando a lista de palavras de alto escore como entrada, e é um modo compacto de armazenar todas essas palavras. Essa busca é rápida, pois requer apenas uma transição por caracter.

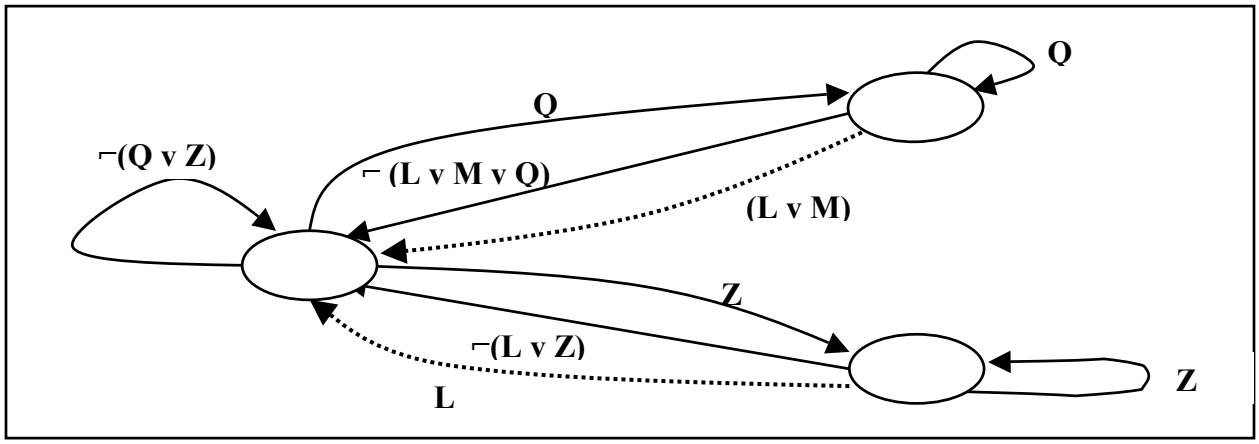

Figura 3.3-21 – BLAST: Autômato finito determinístico para reconhecer a palavras vizinhas QL, QM e ZL. É usado o paradigma Mealy [74](*accept on transitions*) para economizar tempo e espaço. Baseada em [**Craven, 2002**].

2. ***Estender as "sementes" de cada seqüência***:

Cada palavra é estendida tanto para a esquerda quanto para a direita sem introduzir lacunas, o que é mostrado na Figura 3.3-22 a seguir. Em essência, a palavra coincidente é equivalente a um caminho curto na diagonal de um grafo de alinhamento, e é feita uma tentativa para estender o grafo em ambas as direções para, assim, aumentar o escore. Quando uma região que degrada o escore é atingida, a busca é abandonada. Este passo é também muito rápido porque busca apenas um nó no grafo e considera apenas arcos diagonais. Pares de segmentos de escores não suficientemente altos são descartados **[Bioinfo_326, 2002]**.

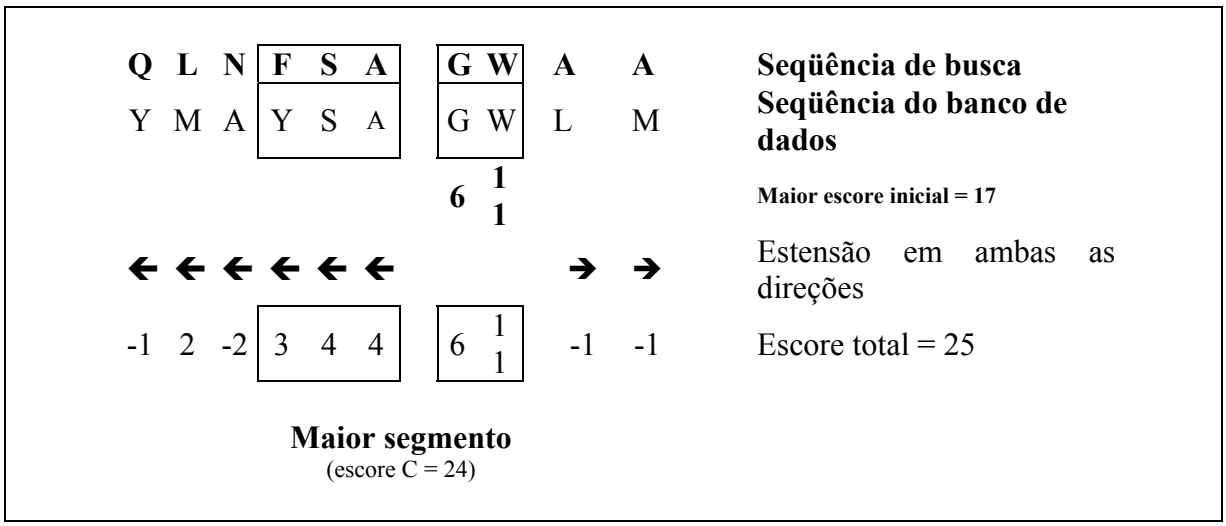
Q L N F S A G W A A
Seqüência de busca
Y M A Y S A G W L M
Seqüência do banco de dados
6 1 1
Maior escore inicial = 17
Estensão em ambas as direções
-1 2 -2 3 4 4 6 1 1 -1 -1
Escore total = 25
Maior segmento
(escore C = 24)

Figura 3.3-22 – BLAST: *Extensão das sementes de cada uma das seqüências***.**
Baseada em [Martins, 2000].

3. ***Estender os alinhamentos de cada segmento restante***:

Começando em um ponto central de cada par de segmentos máximo remanescente, os alinhamentos são estendidos em ambas as direções usando programação dinâmica. Para evitar busca em toda a extensão do grafo de alinhamento, pontos são ignorados onde os escores acumulativos caem mais que um valor X abaixo do mais alto escore conseguido no início deste passo. Cada uma dessas extensões é bastante longa se comparadas àquelas do passo anterior, embora isso seja feito muito menos freqüentemente **[Bioinfo_326, 2002]**.

4. ***Criar uma matriz de escore para as seqüências altamente conservadas***:

Tendo sido rastreado todo o banco de dados, comparam-se as seqüências de alto escore que apresentam identidade com a seqüência de busca para determinar quais delas são altamente conservadas. Uma matriz de escore de posição específica (*PSSM – Position-Specific Scoring Matrix*) é construída utilizando essas informações. Por exemplo, se a seqüência de busca (S1) se identifica fortemente com 10 seqüências do banco de dados (Sn´s), de tal modo que o A de S1 na posição 37 se alinhe com outro A na posição 9 de qualquer das Sn´s, enquanto o A de S1 na posição 119 é alinhado com diversas letras em outras seqüências. Com escores de posições específicas, o primeiro alinhamento pode gerar um alto valor, enquanto o segundo alinhamento do A na posição 119, mesmo sendo com outro A, geraria um valor não muito significativo e comparável a alinhá-lo com qualquer outra letra. Repetidas vezes o banco de dados será rastreado usando essa matriz adaptada para os escores dos alinhamentos. O processo pode ser repetido usando seqüências similares descobertas pelo passo 2 no banco de dados **[Bioinfo_326, 2002]**.

III. Comparação entre BLAST e FASTA

As escolhas de uma matriz de escore, de W (tamanho da palavra) e de T (escore da palavra), em relação a C (escore do segmento), são críticas para a eficiência e efetividade do BLAST. Ao se diminuir T, reduz-se a chance de que o escore de um par de segmentos máximo acima de um valor C seja perdido, mas aumenta o tempo de uso de máquina. Essas escolhas vem sendo intensivamente estudadas de modo empírico e, por isso, os valores “defaults” têm sofrido mudanças ao longo do tempo para se ajustar a cada nova realidade atingida. Atualmente, o valor recomendado de W tem sido na faixa de 3 a 5 para resíduos de aminoácidos e 12 para nucleotídeos **[Gusfield, 1997]** .

Inicialmente relatou-se que o BLAST executaria mais rapidamente que o FASTA e ainda muitas vezes mais rápido que os algoritmos de programação dinâmica para alinhamento local. Entretanto, comparações do BLASTP com versões mais atuais de FASTA mostraram que a diferença de velocidade entre eles é muito sutil e que ambos

continuam significativamente mais rápido que os algoritmos rigorosos, exceto em algumas circunstâncias muito limitadas.

Mesmo sendo os resultados de efetividade biológica (sensitividade e selectividade dos métodos de busca) obtidos por investigações empíricas, o autor afirma que o BLAST é perfeitamente competitivo com o FASTA, embora aquele seja um pouco menos efetivo que esse para identificar importantes identidades, particularmente quando as seqüência possuem espaços ou lacunas. Por isso, há a recomendação de que sejam usados redundamente BLAST e FASTA, o que faz sentido, bem como o faz utilizar o algoritmo Smith-Waterman (alinhamento local) em toda a extensão das duas seqüências localmente similares encontradas por qualquer dos algoritmos heurísticos. E Willian R. Pearson acrescenta: “enquanto BLASTP é efetivo para identificar relações distantes, um alinhamento do tipo Smith-Waterman deve sempre ser usado conjuntamente quando as identidades são analisadas e mostradas”.

3.4 ENTENDENDO UM PROJETO GENOMA

Em relação aos genomas, nos últimos anos houve uma verdadeira explosão desses projetos, cujo objetivo é conhecer a informação contida no DNA das células, que é o que determina as suas possibilidades. Ou seja, é lá que estão escritas as instruções para o funcionamento das células **[Pereira, 2001]**.

Se se considerasse uma célula como um computador, poder-se-ia dizer que a realização de um projeto genoma equivale a vasculhar a máquina em busca dos programas que essa tem instalado no seu hardware. Entretanto, essa identificação não traz informações sobre os programas que efetivamente estão rodando em um determinando momento ou sob uma determinada condição. Por isso, além de descobrir os genes que um determinado organismo possui, um projeto genoma também pretende desvendar funcionamento, expressão e função desses.

Devido às previsões de que o planeta receberá 2,5 bilhões de novos habitantes nos próximos 25 anos e que os países em desenvolvimento necessitarão dobrar sua produção de alimentos, um projeto genoma pode ser, por exemplo, a saída para vencer os fatores biológicos que influenciam negativamente o aumento dessa produção: perdas provocadas

por pragas e doenças. Estima-se que 40% e 20% da produção agrícola em países em desenvolvimento e desenvolvidos, respectivamente, é perdida devido a ação de pragas e patógenos. Desses, um terço pela ação de vírus, fungos e bactérias. Além desses fatores, também existem as barreiras hidrogeográficas (ou fatores físicos, como o suprimento de água doce e a existência de terras agricultu ráveis), contra as quais há pouco a ser feito, no máximo um melhor gerenciamento dos recursos. Quanto aos biológicos, aí sim, há muito a ser feito.

Para os males da explosão demográfica apontados pelo economista e pastor anglicano Thomas Robert Malthus em 1798, no seu famoso livro "População: o primeiro ensaio", a solução também ocorreu pelo auxílio da ciência, embora de um modo mais próprio aos séculos que se seguiram. Segundo ele, a população crescia em progressão geométrica (1, 2, 4, 8, 16, 32, 64...), obviamente com maior rapidez que os meios de subsistência, que cresciam em progressão aritmética (1 2, 3, 4, 5, 6...). Devido ao desequilíbrio entre os recursos naturais e as necessidades da população, **[Araújo, 2002]** a catástrofe malthusiana ocorreu na Irlanda, com a fome provocada pela escassez de batatas no século XIX. E, no século XX, novas catástrofes malthusianas ocorreram na Etiópia e Somália.

Felizmente, as profecias de Malthus ainda estão longe de se concretizarem nos países desenvolvidos, onde houve um aumento populacional, mas também houve aumento da produção devido aos avanços na tecnologia - como adubação, uso de pesticidas e melhoramento genético - e na medicina nos últimos dois séculos. Entretanto, o aspecto malthusiano ainda amedronta os países subdesenvolvidos da África (são 2.250.000 mortos em guerras civis), da Ásia (71,4% da população vive abaixo da linha de pobreza) e da América do Sul (estimou-se mais de 349 milhões de habitantes no ano 2000).

Plantas e patógenos: o paradigma da "Red Queen" e projeto genoma

Na famosa história infantil "Alice no país das maravilhas" (de Lewis Caroll), há uma curiosa passagem em que a personagem corre, corre, corre, mas tudo ao seu redor também corre. Ou seja, é uma corrida para não se sair do lugar. Dessa passagem, surgiu

o paradigma (ou hipótese) da "Red Queen"[xxv], largamente utilizado em perspectiva evolutiva para se comparar o que ocorre entre hospedeiros e patógenos: por mais que os hospedeiros "corram" para buscar novas estratégias para escapar dos seus algozes, os patógenos sempre acabam encontrando formas de superar a resistência conseguida pelos primeiros. É uma corrida sem fim, mas que deve ser disputada, e o papel da ciência nessa corrida é o de prover uma "dianteira" para os hospedeiros **[Pereira, 2001]**.

Com esse objetivo, a ciência apresenta duas necessidades: a) entender os competidores, e aí se encaixa a ciência básica, na qual se inclue o projeto genoma; e b) geração de estratégias capazes de interferir com os competidores, quer seja para prejudicá-los - como o uso de fungicidas, herbicidas e pesticidas - ou para auxiliá-los, como acontece com a aplicação de adubos ou o melhoramento genético. Nesse último caso, incluem-se, além do método convencional**,** os controversos transgênicos, aos quais um projeto genoma é muitas vezes erroneamente associado.

Etapas e técnicas aplicadas

Qual o número de seqüências necessário para se realizar um genoma completo? Não há uma resposta simples para essa pergunta, visto que isso depende da complexidade de cada tipo de genoma, no qual tem-se que considerar o conteúdo de zonas repetitivas, regiões de difícil clonagem, de baixa complexidade, entre outros. Entretanto, convencionou-se considerar que o seqüenciamento de uma quantidade de nucleotídeos que permita cerca de 5X a "cobertura" do genoma é um número com o qual normalmente se atinge acima de 95% de fechamento. Por cobertura, entenda-se o número mínimo teórico de "reads" que seria capaz de fechar um determinado genoma se não houvesse sobreposição entre eles. Por exemplo, um genoma de 2 Mb necessita teoricamente de um mínimo de 4000 reads para cobri-lo (4000 reads x 500 bp = 2.000.000bp) **[Pereira, 2001].**

Para lidar com o genoma, o projeto é dividido em duas grandes etapas: seqüenciamento e anotação. Seqüenciamento é a determinação da ordem de nucleotídeos (DNA ou RNA) ou a ordem de aminoácidos em uma proteína. **[Santos e Ortega, 2001]** A anotação, que é a etapa seguinte, envolve a atribuição de funções e identificação

[xxv] Mais detalhes em http://sunflower.bio.indiana.edu/~clively/Research/Red_Queen%20hyp.html.

de padrões e de genes na seqüência linear do DNA obtida do seqüenciamento. Toda essa informação está disponível nas diferentes ordens e arranjos das seqüências de DNA.

Quanto ao seqüenciamento, três técnicas são empregadas:

a) ***Shotgun* ou "tiro no escuro"**: todo o DNA do organismo é aleatoriamente fragmentado em pequenos pedaços de 1000 a 2000 bp, esses fragmentos abastecem as máquinas automáticas de seqüenciamento e posteriormente são montados (*assembling*) via bioinformática, como se fossem peças de um quebra-cabeça, formando uma seqüência contígua, ou *contig*. As seqüências com sobreposição são ordenadas pelas suas partes comuns de forma a recomporem a seqüência original **[Pereira, 2001]**.

Um problema dessa metodologia é a ordenação de fragmentos em regiões repetitivas. Ou seja, como em cada reação de seqüenciamento obtêm-se a identidade de apenas cerca de 500 bp (denominado "*read*"), se no genoma existem zonas de DNA repetitivo com extensão maior do que essa, torna-se difícil interpretar a localização das seqüências individuais. Para contornar esse problema, é comum, em genomas complexos, a realização de clonagem de fragmentos maiores em vetores do tipos cosmídeos ou cromossomos artificiais de bactérias (BAC) ou leveduras (YAC), que são seqüenciados individualmente e levados para a montagem já como grandes pedaços.

Embora seja o "shotgun" uma estratégia simples, a etapa de montagem de grandes genomas usando essa técnica exige um enorme esforço computacional, como aconteceu com a montagem do genoma humano pela Celera, feita a partir de 26,4 milhões de fragmentos de DNA com tamanho médio de 550 bp exigiu mais de 20.000 horas de processador do mais poderoso computador civil do mundo. Por outro lado, a montagem de pequenos genomas pode atualmente ser realizada por computadores relativamente modestos, de preço aproximado de U$ 10.000,00, utilizando processadores tipo INTEL, sistema Linux. **[FAPESP, 1999]** Foi também empregada pela FAPESP/UNICAMP no seqüenciamento da *Xylella fastidiosa*, a bactéria causadora da Clorose Variegada dos Citros (CVC), praga popularmente conhecida como amarelinho.

A par do procedimento inteiramente *shotgun*, as metodologias convencionais de seqüenciamento utilizam várias etapas de subclonagens

(hierarquia de clonagens) dependentes de mapeamentos diversos para ordenamento das seqüências feitas a partir de clones em plasmídios, que são montados em segmentos contínuos de DNA (contigs), em pedaços cada vez maiores, até o cromossomo completo [Santos e Ortega, 2001]. Vide Figura 3.4-1 a seguir:

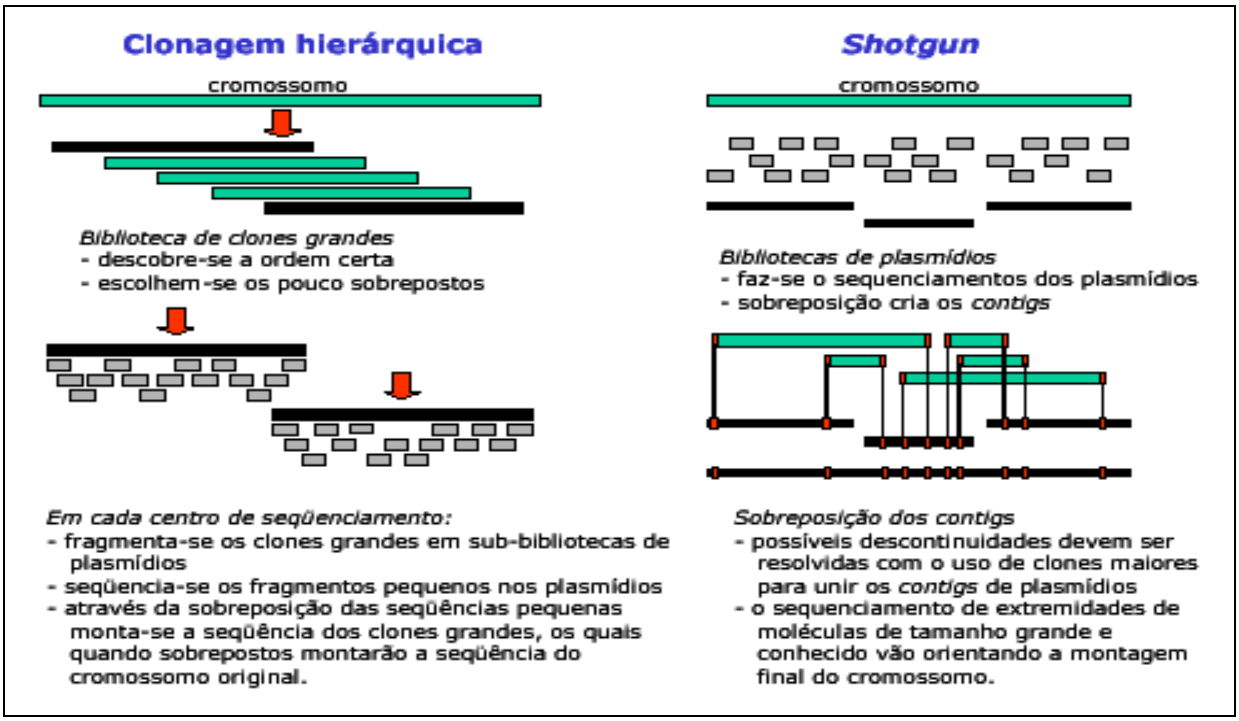
Clonagem hierárquica
cromossomo
Biblioteca de clones grandes
- descobre-se a ordem certa
- escolhem-se os pouco sobrepostos
Em cada centro de seqüenciamento:
- fragmenta-se os clones grandes em sub-bibliotecas de plasmídios
- seqüencia-se os fragmentos pequenos nos plasmídios
- através da sobreposição das seqüências pequenas monta-se a seqüência dos clones grandes, os quais quando sobrepostos montarão a seqüência do cromossomo original.
Shotgun
cromossomo
Bibliotecas de plasmídios
- faz-se o sequenciamentos dos plasmídios
- sobreposição cria os contigs
Sobreposição dos contigs
- possíveis descontinuidades devem ser resolvidas com o uso de clones maiores para unir os contigs de plasmídios
- o sequenciamento de extremidades de moléculas de tamanho grande e conhecido vão orientando a montagem final do cromossomo.

Figura 3.4-1 – Seqüenciamento por clonagem hierárquica e por shotgun.
Fonte: http://www.icb.ufmg.br/~lbem/aulas/grad/tge/bioinfo/bioinfogenomica.pdf

Para executar essa montagem feita pela superposição das seqüências dos vários clones, novas ferramentas da bioinformática foram construídas para análise das milhares de seqüências de DNA geradas pelo seqüenciador automático: PHRED e PHRAP. O PHRED verifica a qualidade do seqüenciamento de cada base das várias seqüências; já o PHRAP faz o alinhamento de todos os clones, construindo uma seqüência contínua, ou *contig*. No final, vários contigs irão compor um grande contig que pode ser a fita de DNA completa de um cromossomo de bactéria, que é na maioria dos casos o seu genoma completo.

Para a montagem final, várias outras ferramentas foram desenvolvidas para manipulação e ordenamento de grandes contigs, bem como a visualização do mapa final com toda a anotação funcional (Ex: Mummy e Assembler do TIGR). Nos eucariotos, cada cromossomo possui uma molécula de DNA e,

como humanos têm 24 tipos de cromossomos (1 a 22, X e Y), deve-se seqüenciar completamente 24 dessas moléculas, avançando-se muitas vezes por longos trechos de DNA repetitivo, que são praticamente impossíveis de seqüenciar com perfeição.

b) **ESTs (*Expressed Sequence Tags)***: esta abordagem deixa de lado a idéia de seqüenciar todo o DNA de um organismo, e busca as regiões da cadeia de nucleotídeos que codificam proteínas. **[FAPESP, 1999]** A técnica funciona de trás para a frente: os pesquisadores partem não do DNA genômico, mas de sua transcrição: as moléculas de RNA mensageiro. A partir delas, usando a enzima transcriptase reversa, sintetizam em laboratório fragmentos de DNA (cDNAs). As seqüências obtidas desses fragmentos recebem o nome de EST, as "etiquetas de seqüências expressas".

O exemplo da nossa espécie ilustra a vantagem de uma abordagem desse tipo: calcula-se que apenas cerca de 3% dos 3 bilhões de pares de bases do genoma humano formam genes, ou seja, comandam a síntese de RNA e das proteínas que regulam a vida. Essa abordagem alternativa, que não exclui a anterior, apareceu em 1991. O objetivo das ESTs é identificar a presença de genes expressos, associando a etiqueta ao gene (e sua função) através um programa tal como o BLAST, que faz busca por homologias.

Um problema dessa metodologia é a dificuldade de se encontrar a condição correta para se conseguir todos os mRNAs produzidos pelo organismo. Isso porque diferentes tecidos expressam diferentes genes sob diferentes condições e mesmo esses são expressos com diferentes intensidades. Assim sendo, projetos desse tipo raramente conseguem identificar todos os genes de um organismo e, na maioria das vezes, genes fortemente expressos são seqüenciados muitas vezes e os de expressão fraca ou transiente freqüentemente não são identificados **[Pereira, 2001]**.

Freqüentemente as seqüências parciais (ESTs) se originam de ambas as extremidades do cDNA, embora alguns projetos prefiram a extremidade 3' - por facilitar a geração de seqüências consenso através do agrupamento de vários ESTs - enquanto outros escolhem a extremidade 5', por estar mais

próxima da região codificadora da proteína, o que facilita a identificação por homologia **[Santos e Ortega, 2001]**.

c) **ORESTES**[xxvi] **(*Open Reading Frames ESTs*)**: é uma técnica desenvolvida no ano de 2000 por Emmanuel Dias-Neto sob a orientação do britânico Andrew Simpson (coordenador do Genoma Câncer/FAPESP), ambos pesquisadores do Instituto Ludwig de São Paulo e co-autores do estudo. O enfoque inovador é permitir o seqüenciamento da região central dos mRNAs, pois se baseia na amplificação de cDNAs por PCR aleatório cujos produtos são utilizados para gerar uma biblioteca. O seqüenciamento dessa biblioteca, contendo fragmentos aleatórios derivados de diferentes regiões de cada mRNA, favorece o reconhecimento da função do transcrito por pesquisa de homologia, pois incorpora mais freqüentemente a ORF no transcrito do que as ESTs convencionais **[Santos e Ortega, 2001]**.

Os ORESTES foram responsáveis pela identificação de 219 novos genes no cromossomo 22 humano, que não haviam sido detectados previamente por outras análises bioinformáticas. Além disso, o agrupamento de seqüências para geração de consensos é facilitado quando são utilizados ESTs convencionais associados aos ORESTES. Esses consensos são importantes, pois muitas vezes contêm toda região codificadora, o que facilita o processo de anotação gênica em eucariotos.

Uma vez feito o seqüenciamento, outra etapa essencial para um projeto genoma é a anotação dessas seqüências. Na maior parte das vezes, a anotação inicial é feita via comparação das seqüências obtidas com os bancos de dados públicos, onde já existem seqüências anotadas, muitas delas fruto de extenso trabalho de bioquímicos que antecederam a biologia molecular. Existem atualmente muitos desses bancos de seqüências "on line", por exemplo o GenBank, que podem ser consultados via programas de comparação específicos do tipo BLAST. Para a maior parte dos projetos genoma, a anotação inicial de seqüências é feita automaticamente usando esses programas de comparação, sem que experimentos de bancada (wet lab) sejam realizados. São os chamados experimentos "in silica" **[Pereira, 2001]**.

[xxvi] ORESTES - mais detalhes em http://www.fapesp.br/genoma391.htm.

Uma questão sensível é definir o momento de se anotar um genoma. No caso de genomas de *shotgun*, principalmente pequenos genomas, é comum se aguardar a finalização do seqüenciamento e montagem para só então se realizar a anotação. Em caso de cDNA, ao contrário, toda seqüência costuma ser imediatamente comparada e anotada. A vantagem do primeiro procedimento é a segurança da anotação. No segundo caso, a vantagem é a possibilidade de rápida identificação de genes relevantes que possam auxiliar no trabalho de bancada.

Voltando à questão do genoma, saber os genes que um determinado organismo possui não é suficiente para entender como esses genes funcionam. Para responder a essa demanda, nos últimos anos um novo tipo de tecnologia tem sido desenvolvida: o **Microarranjo de DNA**, comumente denominada de Chip de DNA, que opera através do princípio de hibridação de moléculas com seqüências complementares. Por essa técnica, um robô imprime ordenadamente em uma lâmina milhares de fragmentos de DNA correspondente à seqüência de genes. A célula a ser estudada é submetida a diferentes condições, por exemplo crescimento em 30^{o}C ou 37^{o}C, e seu mRNA é extraído, sendo que essas moléculas refletem os genes que estão "ligados" na célula naquele determinado momento. Esses conjuntos de mRNAs são transcritos reversamente na presença de nucleotídeos marcados com diferentes substâncias fluorescentes.

Supondo que os cDNAs gerados de mRNAs das células crescidas a 30^{o}C foram marcados com fluorescência verde e os de células crescidas a 37^{o}C marcados com fluorescência vermelha, esses dois tipos de cDNAs marcados são misturados e hibridados contra o DNA impresso na lâmina. As seqüências dos genes para as quais existirem cDNA marcado vão se ligar a esse cDNA e, conseqüentemente, tornar-se-ão pontos fluorescentes na lâmina. A cor de cada ponto vai indicar a situação fisiológica na qual aquele gene foi expresso, e a intensidade do seu brilho vai ser proporcional à sua intensidade de expressão. Existem excepcionais exemplos da utilização dessa metodologia para a investigação do metabolismo de microorganismos como *Saccharomyces cerevisiae*.

3.5 ALINHAMENTO MÚLTIPLO

Entender estrutura, função e evolução de genes é um dos principais objetivos de projetos de seqüenciamento de genoma. As análises de seqüências homólogas é útil não somente para estudar relações evolutivas, mas também para identificar restrições estruturais ou funcionais de DNA, RNA ou proteína **[Bioinfo_326, 2002]**.

Os programas de alinhamento são desenhados para identificar seqüências homólogas distantes baseados nas semelhanças das seqüências. Quando dizemos que duas seqüências são homólogas, ou seja, que elas descendem de um ancestral comum, assumimos uma hipótese para a história evolutiva das seqüências. Um resultado muito interessante dos projetos de seqüenciamento de genomas bacterianos foi a descoberta de que mais da metade dos genes de um genoma partilham de semelhanças em seqüência com outros genomas que divergiram destes há milhares de milhões de anos. Como a maioria dos estudos moleculares e fisiológicos em biologia são feitos com organismos modelos, a análise de semelhança é um formidável utensílio de análise por homologia da função dos genes **[Rocha, 2000]**.

Quando se pretende caracterizar uma família de seqüências que partilham uma mesma atividade biológica, a utilização de alinhamentos de seqüências duas a duas (ou alinhamento *pairwise*) não é satisfatória, pois não providencia uma comparação do conjunto das seqüências. Pode-se estar interessado na visão global de um conjunto de seqüências por diversas razões, dentre as quais duas se destacam: a) as seqüências têm uma história evolutiva comum e a partir delas podemos estudar a história evolutiva das espécies respectivas; b) as seqüências estão relacionadas por uma razão de ordem funcional ou estrutural e o estudo das suas semelhanças permite acrescentar novas informações sobre elas; e c) há certas seqüências que apresentam função similar em várias espécies diferentes e se deseja saber quais partes dessas seqüências são similares e quais são diferentes. O alinhamento múltiplo foi uma das primeiras respostas a estes tipos de problemas e ainda continua a ser a abordagem dominante, pelo menos no que se refere ao estudo da filogenia.

Na verdade, alinhamento múltiplo é uma generalização natural dos casos de alinhamentos que consideram apenas duas seqüências, pois aqui várias seqüências estão

envolvidas e se quer encontrar o melhor caminho para se alinhá-las simultâneamente: seja $s_1,...,s_k$ um conjunto de seqüências do mesmo alfabeto e, pela inserção de espaços nas seqüências envolvidas, todas tornam-se do mesmo tamanho para viabilizar o alinhamento, obedecendo à restrição de que um espaço nunca poderá estar alinhado a outro (Figura 3.5-1). É a tentativa de posicionar os resíduos em colunas que derivam de um resíduo ancestral comum (consenso). As lacunas (gaps) representam inserções ou exclusões nas seqüências. Assim, um alinhamento (*pairwise* ou múltiplo) é o modelo hipotético de mutações (substituições, inserções e exclusões) ocorridas durante a evolução das seqüências **[Meidanis e Setubal, 1997]**.

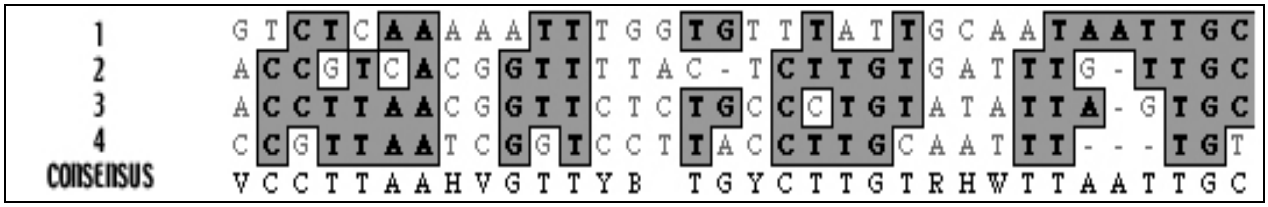

Figura 3.5-1 – Exemplo de alinhamento múltiplo com seqüência consenso.
Fonte: http://www.library.csi.cuny.edu/~davis/Bioinfo_326/lectures/lect8/lect_8.html.

Duas importantes decisões a serem tomadas quanto a este método de alinhamento são: a precisa definição da qualidade do resultado por uma estratégia de escore mais apropriada, por exemplo o sistema SP[75]; e a forma de colocar as seqüências e não simplesmente empilhá-las, o que vem sendo feito usando estruturas em estrela ou árvore.

Os alinhamentos resultantes podem servir a dois propósitos principais: (I) achar regiões de seqüências similares em todas as seqüências que definem um padrão ou domínio conservado de caracteres (consenso); (II) se o alinhamento é particularmente coerente, usar as posições alinhadas para derivar uma possível relação evolutiva entre as seqüências.

O procedimento geral para produzir um alinhamento múltiplo consiste de quatro passos: a) busca para identificar todas as potenciais seqüências homólogas em bancos de dados; b) coletar essas seqüências; c) computar os alinhamentos; e d) checar e editar os alinhamentos.

O alinhamento múltiplo de seqüências é apenas um passo numa metodologia de pesquisa. Tipicamente, após o alinhamento múltiplo das seqüências, estas metodologias encaminham-se para diferentes vias consoante o objetivo do estudo, em particular para

estudos de filogenia ou de funcionalidades. Além disso, há diversos outros usos importantes, a saber **[Bioinfo_326, 2002]**:

a) Demonstração de homologias entre seqüências: baixa similaridade pode não ser considerada significante por um algoritmo de alinhamento *pairwise*. No entanto, essa mesma similaridade pode ser altamente significante se os mesmos resíduos são conservados em várias seqüências distantemente relacionadas. Se as seqüências divergiram muito, a homologia pode não ser reconhecida utilizando apenas alinhamento (vide item "**e**" abaixo e tópico de Predição de estrutura secundária de proteína).

b) Identificação de resíduos altamente conservados que provavelmente correspondam a sítios essenciais para a estrutura ou função da seqüência: pode ser útil para projetar experimentos de mutagêneses[76] que demonstram a importância de sítios, para ajudar a definir os requerimentos da associação de substrato, facilitar o entendimento ou predição de funções mutantes, facilitar a identificação e derivação de bancos de dados de motivos ou domínios (PROSITE, BLOCKS, PFAM etc)

c) Predição ou inferência de função biológica: se um gene é homológo a um outro já caracterizado, é possível inferir a provável função do novo gene a partir da função daquele já conhecido. A estrutura tridimensional de proteínas ou RNA homólogos é freqüentemente mais conservada que a seqüência primária. Com isso, pode-se afirmar que uma estrutura similar implica similaridade de função.

d) Prediçao de estrutura: o uso de alinhamento múltiplo aumenta significativamente a predição de estrutura secundária de proteína. São também comumente usados para modelagem de homologia, isto é, a predição de estrutura de seqüência pela comparação com outras homólogas de estrutura já conhecida. Outro ponto importante é a identificação de sítios covariantes (ou mutações compensatórias) em alinhamentos de RNA ou proteínas, um forte argumento que sugere a interação desses sítios na molécula *in vivo*.

e) Busca por similaridades fracas, mas significantes em bancos de dados de seqüências: a sensitividade das buscas pode ser significativamente melhorada pela atribuição de pesos aos sítios de acordo com o seu grau de conservação.

Assim, os alinhamentos múltiplos de seqüências homólogas podem ser utilizados para gerar matrizes ou perfis específicos de proteína a serem usados por métodos (buscas por perfis ou PSI-BLAST) que identificam membros pouco relacionados de uma família em buscas nos bancos de dados.

f) Projeto de *primers* para a reação em cadeia de polimerase (PCR) visando ampliar ou identificar genes ou proteínas relacionadas: identificar e usar regiões conservadas em um alinhamento para projetar *primers* que isolem um gene ou proteína similar codificando mRNA de um outro organismo.

g) Filogenia molecular: árvores filogenéticas moleculares contam com os alinhamentos múltiplos de proteína ou DNA para inferir eventos de mutação dos quais seja possível hipotéticamente traçar as relações evolutivas entre as seqüências. Tais árvores são úteis para reconstruir a história de espécies ou famílias multigênicas e para identificar eventos de duplicação que distingüem ortólogos e parálogos.

Um ponto importante no caso do alinhamento múltiplo é a validação do alinhamento local ou global. Conforme a Figura 3.5-2, seqüências homólogas podem estar conservadas na totalidade do seu comprimento e, nesses casos, um alinhameto global é o mais indicado para comparar seqüências divergentes. Entretanto, em muitos casos, a homologia pode estar restrita a uma região limitada das seqüências, pois para muitas proteínas, sua organização consiste de uma combinação de módulos discretos que têm sido misturados durante a evolução. Diversos genes codantes são o resultado de recombinações entre diferentes fragmentos de outros genes. A evolução modular de proteínas tem desempenhado importante papel, particularmente nos eucariotos, como resultado do embaralhamento de íntrons e éxons.

Múltiplas cópias de um certo módulo podem ser repetidas dentro de uma seqüência, e um conjunto de módulos pode ocorrer em diferentes posições relativas de genes diversos. Em tais casos, pode não ser possível alinhar seqüências em sua totalidade (um alinhamento global). Conseqüentemente, os alinhamentos poderão ser feitos somente sobre os módulos homólogos (um alinhamento local).

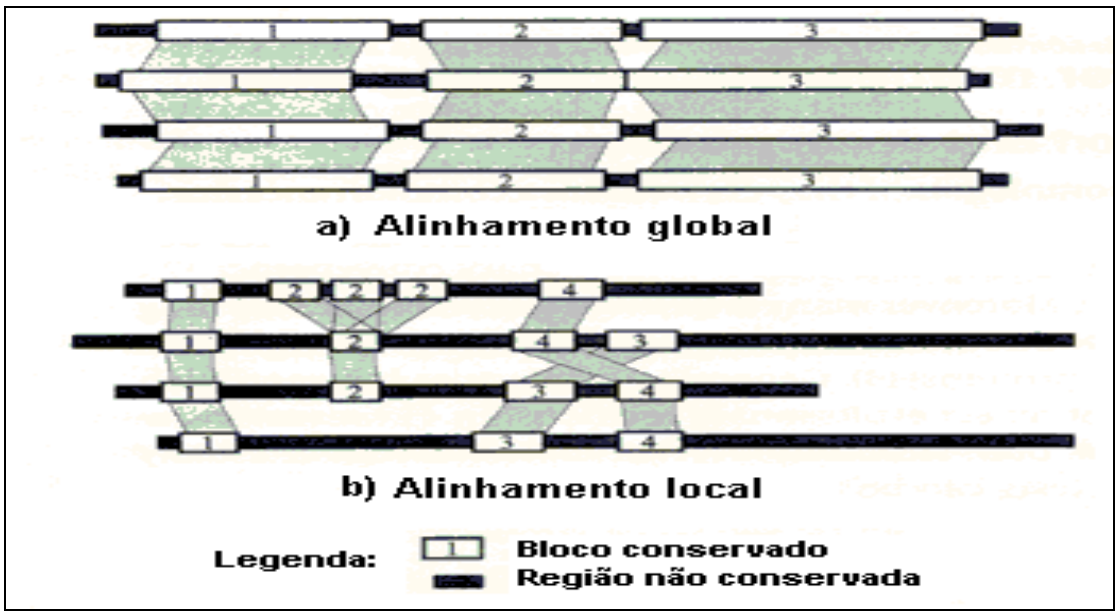

Figura 3.5-2 – Exemplo de alinhamento local e global.
a) regiões conservadas ocorrem na mesma ordem em todas as seqüências e podem ser representadas por um único alinhamennto global. b) Algumas regiões conservadas são duplicadas ou ocorrem em uma ordem diferente ao longo das seqüências. É necessário executar alinhamentos locais para apresentar as similaridades entre todas as regiões conservadas. Fonte: **http://www.library.csi.cuny.edu/~davis/Bioinfo_326/lectures/lect8/lect_8.html**.

O algoritmo de programação dinâmica para alinhamento global é facilmente generalizável ao alinhamento de N seqüências (Kruskal and Sankoff, 1983). No entanto, o tempo e a memória necessários para a sua execução cresce em L* n (L sendo o comprimento característico e n o número de seqüências). Para valores típicos de L de 1000, este tipo de abordagem torna-se impraticável **[Rocha, 2000]**.

Embora diversos algoritmos de alinhamento múltiplo tenham sido desenvolvidos, nenhum deles é ideal. Diferentes algoritmos mostram-se mais apropriados para um alinhamento do que para outro e o problema computacional continua sendo um obstáculo. Assim como acontece com o alinhamento *pairwise* para buscas em banco de dados, heurísticas também são usadas para ganhar velocidade e reduzir o uso de espaço de memória. Conseqüentemente, não existe garantia de se encontrar o alinhamento ótimo na maioria dos casos, mas, mesmo assim, os alinhamentos costumam ser excepcionalmente úteis **[Bioinfo 326, 2002]**. Uma variedade de tipos de algoritmos de alinhamento múltiplo estão disponíveis, entretanto para esta Dissertação será considerado o primeiro item da lista mostrada abaixo:

a) Algoritmos heurísticos que são baseados em um enfoque de alinhamento *pairwise* progressivo. São exemplos: ClustalW (http://www.ebi.ac.uk/clustalw/),

PileUp(CGC-http://www.hgmp.mrc.ac.uk/Courses/gen1day/gcg.practicals.html) e MACAW (disponível via ftp no NCBI ftp://ftp.ncbi.nlm.nih.gov/pubs/schuler/macaw/). O ClustalW é recomendado para alinhamentos locais. A penalidade de lacunas tornam-se mais altas onde as seqüências são mais parecidas, aumentando a probabilidade de encontrar regiões conservadas. Já o PileUp é um programa de alinhamento global e deve ser usado apenas para um conjunto de seqüências parecidas e com o mesmo comprimento aproximado.

b) Aqueles que garantem encontrar o alinhamento ótimo para um certo esquema de escore. Deve-se notar que ótimo é definido aqui em um sentido matemático, que não nercessariamente corresponde ao sentido biológico. Esses algoritmos podem apenas ser usados para um número limitado de pequenas seqüências.

c) Algoritmos heurísticos que constróem um alinhamento global baseado nos alinhamentos locais.

d) Algoritmos heurísticos que constróem alinhamentos múltiplos locais.

e) Algoritmos baseados nos Modelos Ocultos de Markov (HMM[77]).

f) Algoritmos baseados em algoritmos genéticos.

Dentre os algoritmos listados acima, alinhamentos múltiplos progressivos são os métodos mais comumente usados para alinhar seqüências biológicas, pois são rápidos, requerem pouca memória e oferecem boa performance para seqüências homólogas relativamente conservadas. Conforme Figura 3.5-3 a seguir, o alinhamento múltiplo final obtido favorece as seqüências mais parecidas, da seguinte maneira:

a) Computar os escores do alinhamento (ou distâncias) entre todos os pares de seqüências: encontrar quais seqüências são mais similares pela comparação de todas as combinações usando o método de programação dinâmica. As seqüências mais similares são ainda localmente alinhadas utilizando novamente programação dinâmica.

b) Construir uma árvore-guia que reflita as similaridades entre seqüências usando as distâncias do alinhamento *pairwise*: uma seqüência consenso é derivada de cada uma dessas comparações. As seqüências consensos são alinhadas entre si ou com outras similares.

c) Alinhar as seqüências seguindo a árvore-guia: correspondendo a cada nó da árvore o algoritmo alinha dois nós-filhos às duas seqüências ou aos alinhamentos que estejam associados com ele. O processo é repetido começando das folhas (as seqüências) e finalizando na raíz da árvore.

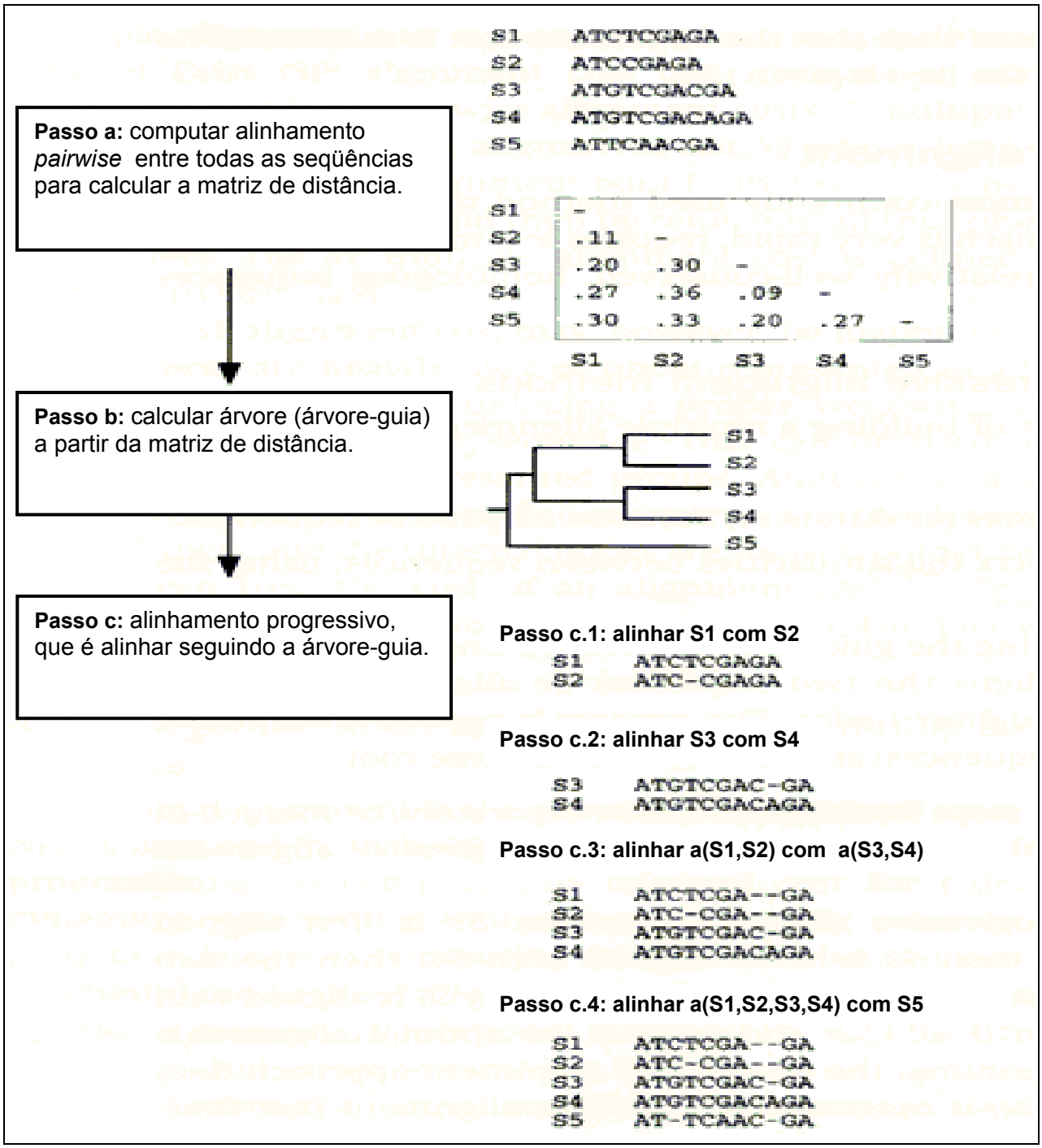

Figura 3.5-3 – O processo de alinhamento progressivo.
Fonte: http://www.library.csi.cuny.edu/~davis/Bioinfo_326/lectures/lect8/lect_8.html.

3.6 PREDIÇÃO DE ESTRUTURA SECUNDÁRIA DE PROTEÍNA

Quando se está lidando com uma seqüência de função desconhecida, a presença de domínios similares em diversas seqüências, também similares, implica função bioquímica ou enovelamento estrutural semelhantes que podem se tornar a base de investigações ou experimentos mais aprofundados, pois um grupo de seqüências similares pode definir uma família de proteína que compartilha uma função bioquímica ou uma origem evolutiva comuns **[Bioinfo 326, 2002]**.

O objetivo de predição de estrutura de proteína é tomar uma estrutura primária (seqüência) e, usando regras derivadas de estruturas conhecidas, predizer a estrutura secundária mais provável a ser adotada individualmente pelos resíduos.

Há alguns atributos a serem assinalados na predição de estrutura secundária de proteína:

a) Deduz as posições mais prováveis de hélices alfa e fitas-beta.

b) Confirma relações funcionais ou estruturais quando a similaridade entre as seqüências é baixa.

c) Determina a direção para uma seleção racional de mutantes específicos a serem utilizados em estudos laboratoriais mais aprofundados.

d) É o fundamento para estudos posteriores baseados em estruturas.

Predição de estrutura secundária vem sendo estudada há quase 25 anos. Os primeiros métodos sofriam pela falta de dados. As predições eram executadas em seqüências simples ao invés de famílias de seqüências homólogas, existiam relativamente poucas estruturas tridimensionais das quais se podiam derivar parâmetros. Provavelmente os métodos iniciais mais famosos são aqueles de Chou & Fasman, Garnier, Osguthorbe & Robson (GOR) and Lim. Embora os autores originalmente atribuíssem alta precisão 70-80%, sob exames cuidadosos os métodos mostraram apenas um nível de precisão entre 56% e 60%[xxvii] **[Russell, 2002]**.

A disponibilidade atual de largas famílias de seqüências homólogas revolucionou a predição de estruturas secundárias. Métodos tradicionais, quando aplicados `a família de

[xxvii] Kabsch, W. & Sander, C. (1983). *FEBS Letters,* **155**, 179-182.

proteínas, provaram muito mais exatidão. A combinação de seqüência de dados com sofisticadas técnicas computacionais, tais como rede neurais, tem levado a índices de precisão que ultrapassam 70%.

Segundo **[Le Novère, 2002],** os diferentes enfoques para predição de estrutura podem ser divididos em três gerações (Tabela 3.6-1):

Apenas os trabalhos originais e as implementações mais recentes são apresentadas aqui.

	Estatística dos resíduos simples[xxviii]	**Regras explícitas**	**Vizinho mais próximo**[xxix]	**Predição baseada em redes neurais**[xxx]
Primeira geração (as informações vêm de simples resíduos e de uma seqüência)	Chou and Fasman 1974 GOR1[xxxi] 1978	Lim 1974		
Segunda geração (Interações locais são levadas em conta)	GOR3[xxxii] 1987		Levin et al 1986 Nishikawa and Ooi 1986	Holley and Karplus 1989 Qian and Sejnowski 1988
		PREDATOR[xxxiii] 1996	Yi and Lander [xxxiv]1993	

[xxviii] Garnier-Osguthorpe-Robson: usa tabela de propensões calculadas primariamente de estruturas determinadas por cristalografia de raio-x. Há uma matriz para cada tipo de característica: hélice-alfa, fita-beta, turn e coil. A característica de mais alto escore é determinada para cada localização.

[xxix] A idéia básica desse enfoque é a predição do estado da estrutura secundária do resíduo central do segmento de teste, baseado na estrutura secundária de segmentos similares de proteínas com estrutura tridimencional já conhecida. A informação vinda de diferentes modelos é medida de acordo com a sua similaridade com o segmento de teste. NNSSP é um aprimoramento do algoritmo projetado por Yi and Lander, 1993.

[xxx] Uma rede neural com múltiplas camadas é apresentada com estruturas e seqüências conhecidas. A rede é treinada até que ela possa predizer as estruturas dadas aquelas seqüências. A camada de entrada é uma seqüência primária. A camada de saída é a probabilidade de que um resíduo de seqüência possa formar uma estrutura particular. A camada oculta (intermediária) é onde o atual aprendizado acontece: um conjunto de treinamento é provido com todas as seqüências de estruturas tridimensionais conhecidas; e o treinamento involve análise de resíduos particulares e suas probabilidades de contribuirem para uma certa estrutura em diferentes contextos. Diferentes redes podem ser criadas para tipos diversos de proteínas. Vide http://www.library.csi.cuny.edu/~davis/Bioinfo_326/lectures/lect11_12/lect_12.html.

[xxxi] Garnier J, Osguthorpe DJ, Robson B (1978) Analysis of the accuracy and implications of simple methods for predictiong the secondary structure of globular proteins. *J Mol Biol*, 120 : 97-120.

[xxxii] Gibrat JF, Robson B, Garnier J (1987) Further developments of protein secondary structure prediction using information theory. New parameters and consideration of residue pairs. *J Mol Biol*, 198 : 425-443.

[xxxiii] Frishman D, Argos P (1996) Incorporation of non-local interactions in protein secondary structure prediction from the amino-acid sequence. *Prot Eng* 9 :133-142. Frishman D, Argos P (1997). Seventy-five percent accuracy in protein secondary structure prediction. *Proteins* 27 : 329-335.

[xxxiv] Yi TM, and Lander S (1993) Protein secondary structure prediction using neirest-neighbor methods. *J Mol Biol*, 232 :1117-1129.

Terceira Geração (Informações vinda de seqüências homólogas são incorporadas)	Zvelebil et al 1987	NNSSP[xxxv] 1995	
	DSC[xxxvi] 1996		PHD[xxxvii] 1993 Jnet[xxxviii] 1999 PSIPRED[xxxix] 1999 NNPREDICT[xl] 1996 SOPMA[xli] 2002

Tabela 3.6-1 – As 3 gerações de enfoques para predição de estrutura secundária de proteína.

Alguns autores, inclusive **[Le Novère, 2002],** aconselham a seguinte prática para lidar com a diversidade de algoritmos de predição de estrutura:

- **a)** Evite o algoritmo Chou and Fasman.
- **b)** Observe a precisão dos algoritmos em *benchmarks* padronizados e situações da vida real.
- **c)** Use métodos baseados em alinhamentos múltiplos.
- **d)** Verifique cuidadosamente os alinhamentos para evitar redundâncias.
- **e)** Use vários métodos independentes e de exatidão similar.
- **f)** Em caso de desacordo, os mais confiáveis são: PHD, Jnet e PSIPRED.

A seguir, a estrutura tridimensional da proteína Flavodoxina (*Anacystis nidulans*)[xlii] (Vide Entrez Structure; PDB = 1CZU; PDBsum) (Figura 3.6-1) e um exemplo de predição de sua estrutura secundária pelo software SOPMA (Figuras 3.6-2 e 3.6-3):

[xxxv] Salamov AA, Solovyev VV (1995) Prediction of protein secondary structure by combining nearest-neighbor algorithms and multiple sequence alignment. *J Mol Biol*, 247 : 11-15.
[xxxvi] King RD, Sternberg MJE (1996) Identification and application of the concepts important for accurate and reliable protein secondary structure prediction. *Prot Sci*, 5 : 2298-2310.
[xxxvii] Rost B, Sander C (1993) Prediction of protein secondary structure at better than 70% accuracy. *J Mol Biol* 232 : 584-599. Rost B, Sander C (1994) Combining evolutionary information and neural networks to predict protein secondary structure. *Proteins*, 19 : 55-72.
[xxxviii] Cuff J. A. and Barton G. J. Application of Enhanced Multiple Sequence Alignment Profiles to Improve Protein Secondary Structure Prediction, submited (1999).
[xxxix] Jones, D. T. (1999) Protein secondary structure prediction based on position-specific scoring matrices. J. Mol. Biol. 292: 195-202.
[xl] http://www.cmpharm.ucsf.edu/~nomi/nnpredict.html.
[xli] [xli] NPS@: Network Protein Sequence Analysis. TIBS 2000 March Vol. 25, No 3 [291]:147-150. Combet C., Blanchet C., Geourjon C. and Deléage G. http://npsa-pbil.ibcp.fr/cgi-bin/secpred_sopma.pl.
[xlii] Flavodoxina - é uma toxina de fungos.

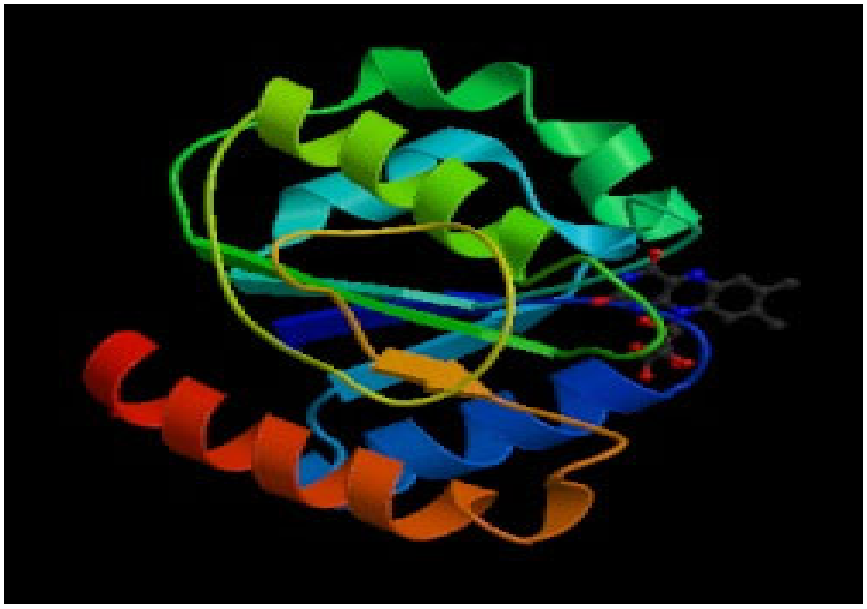

Figura 3.6-1 – A estrutura tridimensional da proteína Flavodoxina.
Fonte: http://www.library.csi.cuny.edu/~davis/Bioinfo_326/lectures/lect11_12/1CZU.jpg.

SOPMA result for : UNK_185240

Abstract Geourjon, C. & Deléage, G., SOPMA: Significant improvement in protein secondary structure prediction by consensus prediction from multiple alignments., Cabios (1995) 11, 681-684
View SOPMA in: [MPSA (Mac, UNIX) , About...] [AnTheProt (PC) , Download...] [HELP]

```
          10         20         30         40         50         60         70
           |          |          |          |          |          |          |
AKIGLFYGTQTGVTQTIAESIQQEFGGESIVDLNDIANADASDLNAYDYLIIGCPTWNVGELQSDWEGIY
ceeeeeecccccchhhhhhhhhhhhttcchheehhhhhhcchhhhhhhheeeeecccccttccccchhhhh
DDLDSVNFQGKKVAYFGAGDQVGYSDNFQDAMGILEEKISSLGSQTVGYWPIEGYDFNESKAVRNNQFVG
hhhhhhccttceeeeeecccccchhhhhhhhhhhhhhhhhhttcceeeeccctteccccchhhhcttceee
LAIDEDNQPDLTKNRIKTWVSQLKSEFGL
eeecccccccchhhhhhhhhhhhttttc
```

Sequence length : 169

SOPMA :

Alpha helix	(Hh)	:	72		is 42.60%
3_{10} helix	(Gg)	:	0	is 0.00%	
Pi helix	(Ii)	:	0	is 0.00%	
Beta bridge	(Bb)	:	0		is 0.00%
Extended strand	(Ee)	:	29		is 17.16%
Beta turn	(Tt)	:	15	is 8.88%	

Figura 3.6-2 – Exemplo de predição de estrutura secundária de proteína. (parte 1 / 2).
Fonte: http://npsa-pbil.ibcp.fr/cgi-bin/npsa_automat.pl?page=/NPSA/npsa_sopma.html.

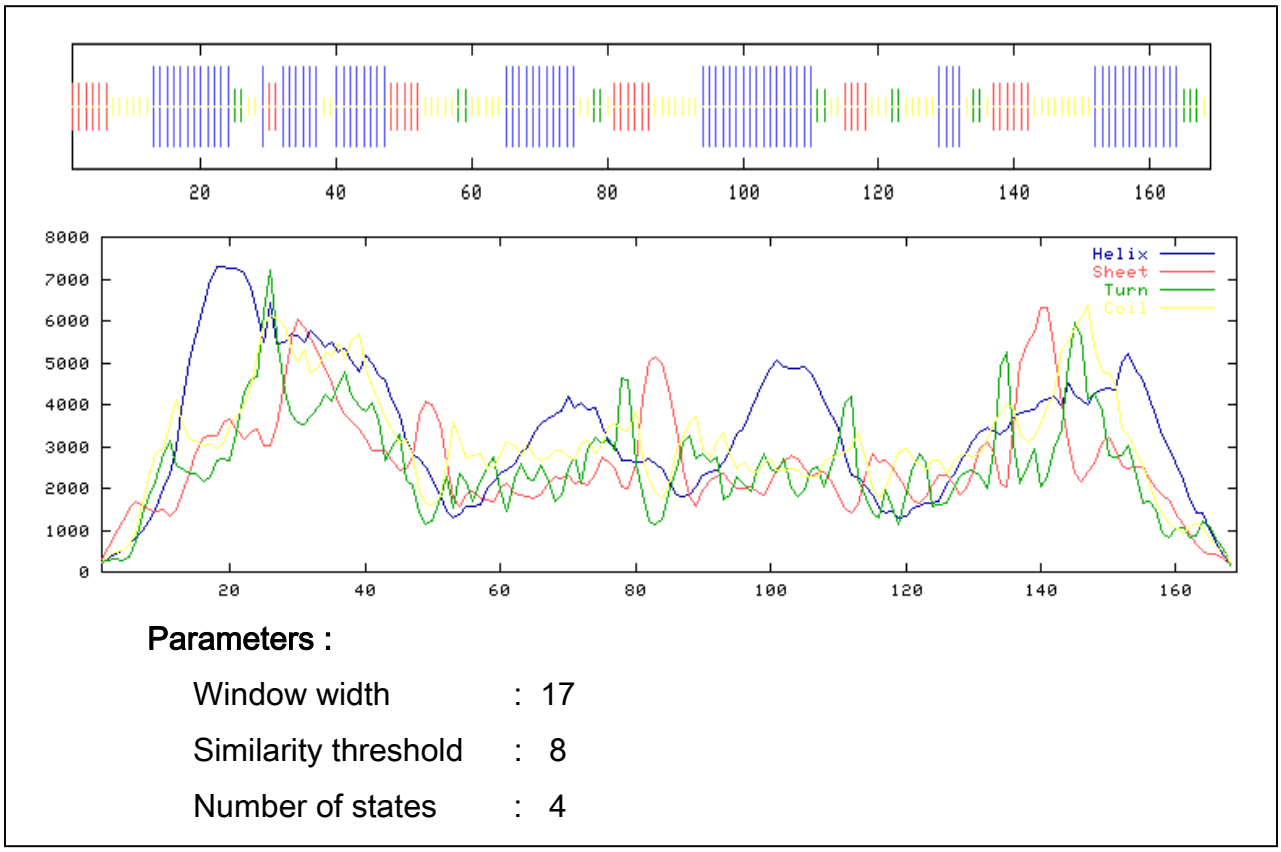

Figura 3.6-3 – Exemplo de predição de estrutura secundária de proteína. (parte 2 / 2).
Fonte: http://npsa-pbil.ibcp.fr/cgi-bin/npsa_automat.pl?page=/NPSA/npsa_sopma.html.

3.7 DERIVAÇÃO DE ÁRVORE FILOGENÉTICA

Tradicionalmente, a classificação de espécies têm sido determinada com base em morfologia comparativa de organismos. Outras características fisiológicas, anatômicas e comportamentais também têm sido usadas. Com isso, os biólogos examinariam uma larga variedade de tratos e fariam julgamentos intuitivos sobre o grau de relacionamento entre os vários grupos de organismos que estivessem estudando. Isso ocorre porque a hipótese primária em análises filogenéticas é que todos os organismos são atualmente relacionados, e derivam de um ancestral comum em algum ponto do passado. O problema principal é determinar quão próxima ou distante é essa relação e que árvore evolucionária descreve como os organismos atuais evoluíram (a árvore verdadeira) **[Bioinfo_326, 2002]**.

Um dos objetos de estudo desta Dissertação é exatamente o ramo da filogenia denominado filogenia molecular, que é a inferência de linhas de ancestralidade para

organismos baseada em seqüências de DNA, RNA ou proteína. Comparações desses tipos de seqüências podem revelar diversas informações sobre a relação entre organismos e que não poderiam ser inferidas pela morfologia ou outras características. E a razão é bastante simples: quanto mais mutações são requeridas para transformar uma seqüência em outra, menos relacionadas são as seqüências e menor a probabilidade de que compartilhem uma recente seqüência ancestral comum.

A filogenia molecular é uma das áreas da evolução molecular que tem tido muito interesse nos últimos anos, principalmente pelo fato de que muitas relações filogenéticas são impossíveis de serem descobertas de outra forma. Além disso, o interesse em biologia da conservação está em alta e através de dados de filogenia molecular é possível descobrir se determinadas populações caracterizam unidades diferentes de significância evolutiva e contribuem para o legado evolutivo de uma espécie, devendo ser conservadas em prioridade **[Santos, 2000]**.

Os principais tipos de seqüências utilizados são: RNA ribossômico, DNA nuclear (codante ou não codante), DNA mitocondrial e seqüências de proteína. Quanto a seqüências de DNA ou proteína, algumas vantagens são apontadas **[Bioinfo 326, 2002]**:

a) Vantagens de DNA: em regiões codantes, mudanças sinônimas (mesmo aminoácido codificado) levam a alterações que podem ser informativas em nível de DNA, mas não de proteína. Esse tipo de seqüência também pode ser de regiões não codantes. Além disso, podem também ser usadas para organismos muito relacionados ou altamente divergentes, dependendo dos marcadores[78] utilizados.
b) Vantagens de proteína: vinte aminoácidos comparados a quatro nucleotídeos permitem o melhor sinal quanto a ruído, numa proporção de 5% contra 25% para DNA. As propriedades dos aminoácidos são essenciais para a função ou estrutura da proteína resultante; já o DNA não é estruturalmente importante. Podem ser usados principalmente organismos que apresentem alta divergência, por exemplo: levedura e vertebrados.

A cladística e a fenética são duas escolas sistemáticas que possuem idéias diferentes sobre qual é o melhor método de diferenciar, classificar e calcular a divergência evolutiva entre organismos **[Santos, 2000]**.

A escola feneticista diz que uma classificação será mais informativa se for baseada na similaridade total entre as espécies, medida por tantas características quanto possível, sejam elas moleculares, fenotípicas ou anatômicas. Os feneticistas desenvolveram métodos matemáticos elaborados para agrupar espécies com relação à sua similaridade global e mostrar essas características num gráfico chamado fenograma. Quanto maiores forem as características comuns nas espécies, maior será o nível em que elas serão agrupadas no fenograma. Entretanto, esse gráfico pode não representar a filogenia verdadeira, pois ele não leva em consideração as taxas diferenciais de evolução.

A escola cladística, que é mais aceita, argumenta que uma classificação deve expressar as relações de ramificação entre as espécies, sem se importar com o grau de similaridade ou diferença. Para um cladista, é interessante notar os caminhos evolutivos seguidos pelos grupos e considera-se que todos os táxons[79] possuem uma origem monofilética e devem incluir todos os descendentes de um ancestral particular. Um gráfico que expresse as relações cladísticas entre grupos é chamado de cladograma, o qual consiste em uma árvore filogenética com uma raiz e os organismos separados por bifurcações.

Para este trabalho, alguns dos principais conceitos sobre árvore filogenética são (Figura 3.7-1):

a) **Nós** : representam unidades taxonômicas.

b) **Galhos**: cada galho conecta nós adjacentes. Definem as relações entre os nós em termos de descendência e ancestralidade.

c) **Tamanho dos galhos**: freqüentemente representa o número de mudanças que ocorrem em relação ao último nó. Às vezes, o tamanho é irrelavante e a árvore representa apenas a ordem de evolução.

d) **Topologia**: padrão de ramificação dos nós e galhos.

e) **Nós internos**: Os nós internos representam unidades ancestrais.

f) **Nós externos**: representam as unidades taxonômicas que estão sendo comparadas. Referem-se a organismos vivos.

g) **OTUs** (*Operational Taxonomic Units* ou Unidades Taxonômicas Operacionais):nível taxonômico de amostragem selecionado pelo usuário para

ser usado no estudo, tal como indivíduos, populações, espécies, gênero, proteínas, genes etc.

h) **Raiz**: um nó ancestral comum de todos os demais nós. Uma árvore pode conter ou não raiz.

i) **Escala de distância**: representa o número de diferenças entre organismos ou seqüências.

j) **Clado**: um grupo de organismos que descendem de um ancestral comum e todas os organismos derivados são incluídos, o que é também chamado de grupo monofilético. É o único tipo de agrupamento aceito pelos cladistas, ou seja, não são aceitos os grupos parafilético[80] e polifilético[81].

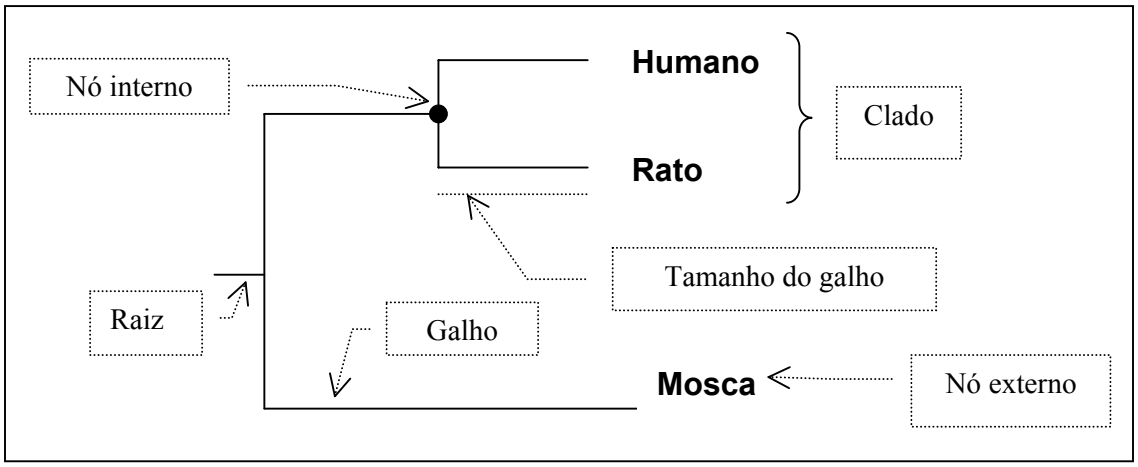

Figura 3.7-1 – Esquema de uma árvore filogenética.
Baseada em http://www.ncbi.nlm.nih.gov/About/primer/phylo.html.

Os principais tipos de árvores são **[Bioinfo_326, 2002]**:

a) árvore com raiz (*rooted tree*): aquela em que um nó especial indicando o ancestral comum para todas as OTUs está presente (a raiz).

b) árvore sem raiz (*unrooted tree*): aquela em que a posição do ancestral comum não está especificada. Mostra apenas as relações evolucionárias entre os organismos, sem mostrar a direção do processo.

c) árvore real (*true tree*): representa o caminho evolucionário atual pelo qual o conjunto de organismos foi criado.

d) árvore inferida (*inferred tree*): é criada por programas de análise evolucionária.

e) árvore de espécie (*species tree*): representa a evolução de um grupo de espécies.

f) árvore de gene (*gene tree*): é construída a partir de um gene de cada espécie. É tida como uma representação mais acurada e menos ambígua de uma árvore de espécie do que aquela obtida apenas por comparações morfológicas.

Quanto à representação de uma árvore, além do cladograma (modo gráfico), há um outro tipo simplificado e em modo texto, que recebe o nome de aninhamento de parênteses, conforme Figura 3.7-2:

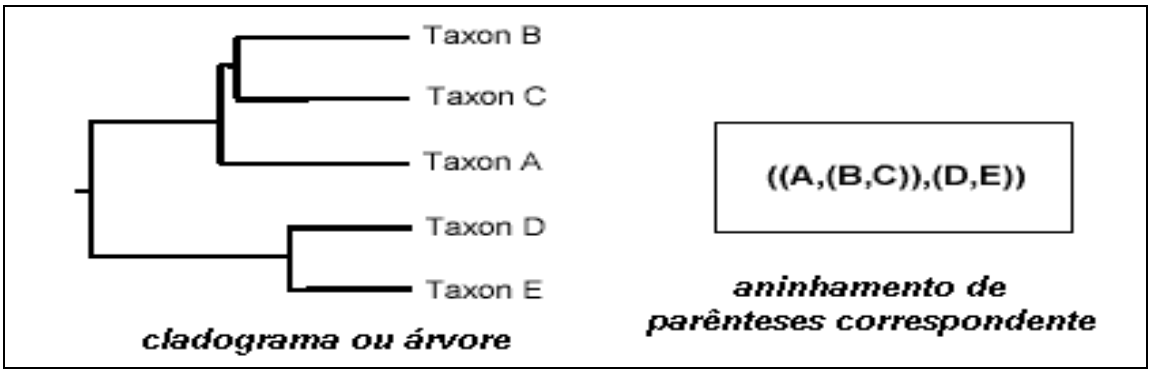

Figura 3.7-2 – Exemplo de cladograma e aninhamento de parênteses correspondente.
Fonte: http://www.library.csi.cuny.edu/~davis/Bioinfo_326/lectures/lect14/lect_14.html.

Inferências filogenéticas são procedimentos de estimativas. Assim, existe uma série de algoritmos que podem ser aplicados e que são classificados em duas categorias (Figura 3.7-3): critérios qualitativos, baseados em otimização dos dados; e critérios quantitativos, baseados na distância entre os dados **[Fernandes-Matioli, 2001]**.

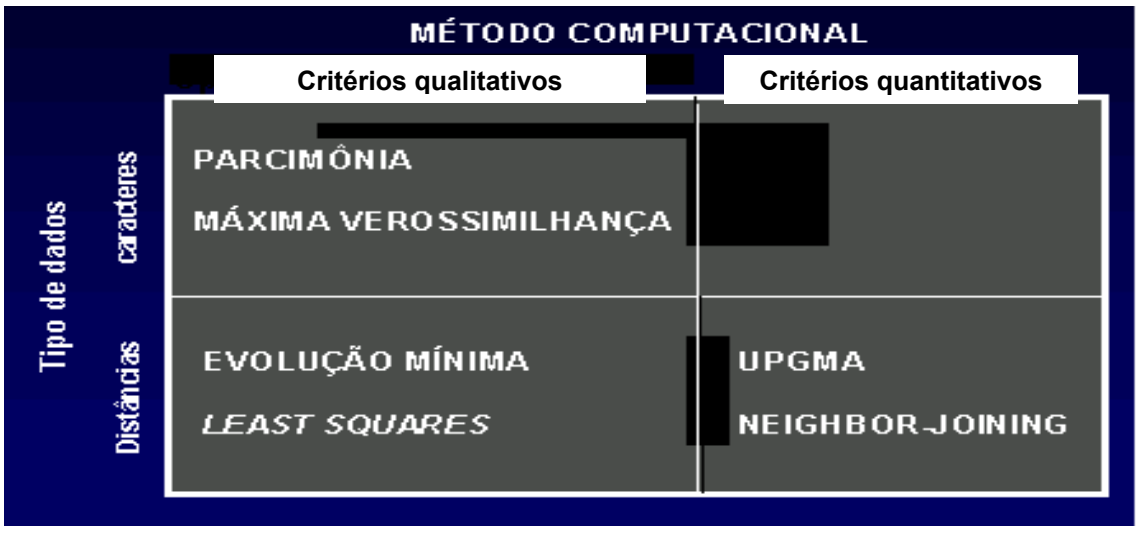

Figura 3.7-3 – Quadro explicativo dos métodos utilizados na construção de árvores filogenéticas moleculares.
Fonte: http://www.icb.ufmg.br/~lbem/aulas/grad/evol/evolmol/.

Nos métodos de distância, as diferenças entre duas seqüências são reduzidas a uma só variável (número de diferenças) e suas relações evolutivas não são consideradas. Nesse método, primeiramente calcula-se a distância, para, a seguir, reconstruir a árvore filogenética, utilizando um algoritmo específico. Existem diferentes modelos que podem ser utilizados para a construção de uma matriz de distância. A partir dessa matriz, será reconstruída a árvore com um dado algoritmo, sendo os dois mais utilizados:

a) UPGMA (***U**nweighted **P**air **G**roup **M**ethod with **A**rithmetic mean*): utiliza um algoritmo de organizações seqüenciais, nos quais as relações topológicas são identificadas por ordem de similaridade e a árvore filogenética é construída passo-a-passo. Primeiro deve-se identificar, dentro de várias OTUs, as duas que são mais similares e tratá-las como uma única, chamada de OTU composta. A partir daí, são observados os outros grupos de OTUs e é identificado o próximo par com maior similaridade, que é novamente arranjado e assim por diante, até que sobrem apenas duas OTUs (Figura 3.7-4).

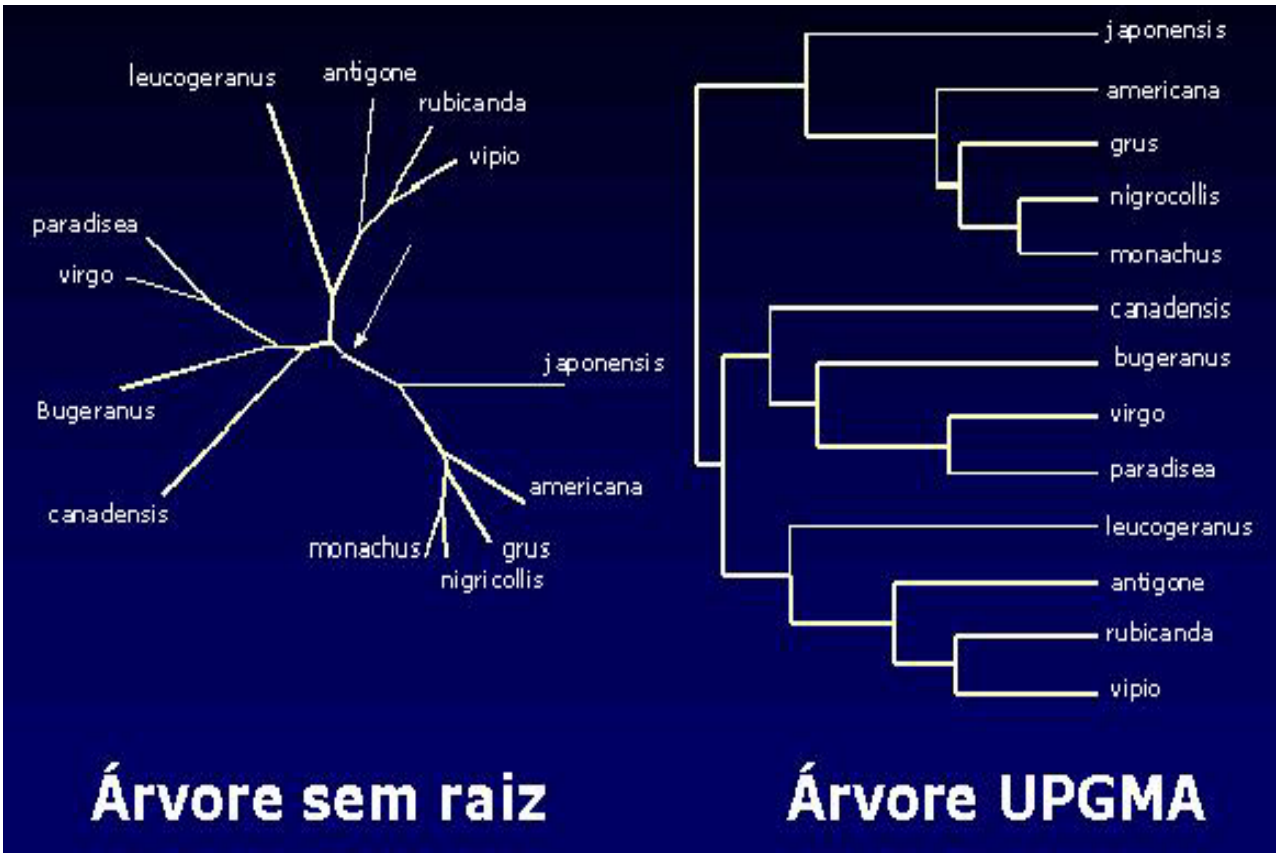

Figura 3.7-4 – Exemplo de árvore sem raiz e árvore UPGMA correspondente.
Fonte: http://www.icb.ufmg.br/~lbem/aulas/grad/evol/evolmol/.

b) *Neighbour-Joining:* assume aditividade, isto é, dada uma árvore, os seus comprimentos são aditivos se a distância entre quaisquer pares de elementos é a soma dos comprimentos dos sub-percursos que os unem (Durbin et al. 1998). Se pudéssemos determinar a partir de uma tabela de distâncias os pares de elementos mais próximos, a árvore seria construída de forma imediata e com comprimentos aditivos exatos. No entanto, isto geralmente não é possível porque os comprimentos dos ramos são diferentes

Nos critérios qualitativos, as diferenças entre as moléculas são consideradas como uma série de variáveis descontínuas. Os dois métodos mais empregados, embora não sejam os únicos, são :

a) Máxima Parcimônia (MP): a hipótese mais simples deve ser a escolhida dentre todas as hipóteses possíveis de reconstrução filogenética. Ou seja, a árvore que apresentar o menor número de passos (passos = mudanças de estado de caráter) será a árvore mais parcimoniosa e será a escolhida para explicar um determinado conjunto de dados. Existem, no entanto, alguns problemas e limitações, pois um determinado conjunto de dados pode gerar mais de uma árvore igualmente mais parcimoniosa, pois os dados estudados podem ser acomodados de formas diferentes e apresentar o mesmo número menor de passos.

b) Máxima Verossimilhança (MV): baseia-se em modelos evolutivos explícitos de substituição de nucleotídeos que são avaliados quanto à sua probabilidade de explicar um conjunto de dados de forma que reflita a história evolutiva mais verossímil. O modelo que apresentar o melhor valor de verossimilhança, que, por questões operacionais, é dado em forma logarítmica, será o escolhido como base para a reconstrução da árvore.

Tanto para MP quanto para MV, existem dois tipos de algoritmo de busca da melhor árvore: a) busca exaustiva, em que todas as possibilidades de topologias são verificadas; e b) busca heurística, em que apenas um subconjunto de árvores probabílisticamente mais prováveis é examinado. Essa última, em termos práticos, reduz sensivelmente o tempo de processamento computacional que, em alguns casos, dependendo do número de seqüências comparadas, pode durar dias e até meses!

Por meio de uma consulta à internet[xliii], foram detectados 194 pacotes de análise filogenética e 16 servidores de livre acesso. Os softwares podem ser consultados e acessados pelo método empregado, pelo sistema operacional no qual são executados, pelos tipos de dados que analisam etc. Alguns dos principais pacotes são: PHYLIP[xliv], PAUP*[xlv], MEGA[xlvi], TREECON[xlvii] e GeneTree[xlviii] .

[xliii] Mais detalhes em http://evolution.genetics.washington.edu/phylip/software.html.
[xliv] PHYLIP - http://evolution.genetics.washington.edu/phylip.html.
[xlv] PAUP - http://paup.csit.fsu.edu/index.html.
[xlvi] MEGA- http://www.megasoftware.net/.
[xlvii] TREECON- http://rrna.uia.ac.be/dcse/help/treecon.html.
[xlviii] GeneTree- http://taxonomy.zoology.gla.ac.uk/rod/genetree/genetree.html.

4. METODOLOGIA

Dentre as metodologias utilizadas nesta Dissertação, destacam-se:

4.1 MODELAGEM ORIENTADA A OBJETOS

[Booch et al, 2000] A atividade de modelagem de um software pode ser analisada pelo enfoque tradicional ou pelo enfoque contemporâneo. Para o primeiro enfoque, o desenvolvimento adota a perspectiva de um algoritmo e o principal bloco de construção do software é o procedimento ou função. A atenção dos desenvolvedores fica voltada para questões referentes ao controle e à decomposição de algoritmos maiores em algoritmos menores. Com isso, o sistema resultante pode ser instável e de difícil manutenção para contemplar alterações de requisitos ou demanda de novas funcionalidades.

O enfoque contemporâneo, no entanto, adota uma perspectiva orientada a objetos, na qual o principal bloco de construção de todos os sistemas de software é o objeto ou a classe. De maneira simplificada, um objeto é alguma coisa geralmente estruturada a partir do vocabulário do espaço do problema ou do escopo da solução; e uma classe é a descrição de um conjunto de objetos comuns. Todos os objetos possuem uma identidade única, um estado (com atributos associados) e um comportamento (com operações ou métodos associados). Essa perspectiva é superior para a criação de arquiteturas flexíveis, inclusive no caso de sistemas que poderão conter grandes bancos de dados ou vários componentes computacionais.

E é nesse contexto de mudança do paradigma tradicional para o orientado a objetos que está inserida a Linguagem Unificada de Modelagem (UML – Unified Modeling Language): padrão de fato adotado pelo OMG (*Object Management Group o*u Grupo de Gerenciamento de Objetos) em janeiro de 1997, em resposta à sua própria solicitação de propostas para uma linguagem-padrão de modelagem. A UML é, na verdade, resultado da unificação de três métodos bastante utilizados na época : Booch (de Grady Booch da Rational Software Corporation), OOSE- *Object-Oriented Software Engineering* (de Ivar Jacobson da Objectory) e OMT-*Object Modeling Technique* (de James Rumbaugh da General Electrics).

A UML é uma linguagem gráfica para visualização, especificação, construção e documentação de artefatos de sistemas complexos de software. Para cobrir todas as etapas do desenvolvimento (levantamento de requisitos, análise, projeto, implementação e testes) são disponibilizados nove diagramas : diagrama de caso de uso, diagrama de atividades, diagrama de seqüência, diagrama de classes, diagrama de colaboração, diagrama de objetos, diagrama de gráficos de estados, diagrama de componentes e diagrama de implantação **[Booch et al, 2000]**.

Para esta Dissertação, nem todos os diagramas foram utilizados, privilegiando-se os seguintes:

a) **Diagrama de casos de uso**: mostra um conjunto de casos de uso e atores e seus relacionamentos. Um caso de uso é uma descrição de um conjunto de seqüências de ações que um sistema executa para produzir um resultado de valor observável por um ator. O ator respresenta um papel que um ser humano, um dispositivo de hardware ou até outro sistema desempenha com relação ao sistema sendo modelado.

Neste trabalho, foram aplicados alguns desses diagramas para ilustrar a visão estática e a organização do comportamento das principais funcionalidades.

b) **Diagrama de atividades**: mostra o fluxo de controle de uma atividade para outra. Uma atividade é uma execução em andamento não-atômica em uma máquina de estados e efetivamente resulta em alguma ação, que pode provocar mudança de estado do sistema ou retorno de um valor. Esse diagrama, caso não existam atividades síncronas, é essencialmente um fluxograma.

Neste trabalho, foi utilizado um diagrama deste tipo para representar a visão dinâmica de cada uma das diversas funcionalidades do Sistema.

c) **Diagrama de interação**: seus subtipos são os diagramas de seqüências e os de colaboração. Exibem uma interação, consistindo de um conjunto de objetos e seus relacionamentos; e as mensagens que podem ser trocadas entre esses objetos. Um diagrama de seqüência tem ênfase na ordenação temporal das mensagens; já o diagrama de colaboração, este mantém a ênfase na organização estrutural dos objetos que enviam e recebem mensagens.

Neste trabalho, foram utilizados diagramas de seqüências para representar os diversos e possíveis cenários dos fluxos básico e alternativo de cada funcionalidade do Sistema.

d) **Diagrama de classes**: mostra um conjunto de classes, interfaces e colaborações e seus relacionamentos.

Neste trabalho, foram utilizados vários diagramas de classes para fazer a modelagem da visão estática do Sistema, pois oferecem principalmente suporte para os requisitos funcionais (os serviços a serem fornecidos aos pesquisadores, os usuários finais). Prestaram-se a auxiliar: na determinação do domínio e seu vocabulário, suas abstrações e responsabilidades; na modelagem das colaborações simples entre as classes de interface com o usuário, de controle e persistentes; e, em muitos casos, na modelagem do esquema lógico das classes persistentes (banco de dados).

4.2 RUP

Um processo é um conjunto de passos parcialmente ordenados com a intenção de atingir uma meta. Na engenharia de software, sua meta é entregar, de maneira eficiente e previsível, um produto de software capaz de atender às necessidades de seu negócio **[Booch et al, 2000]**.

A UML é amplamente independente de processo, significando que é possível utilizá-la com diferentes processos de engenharia de software. O RUP (***R**ational **U**nified **P**rocess* ou Processo Rational Unificado) é um desses processos que se alinha aos propósitos da UML, o que resultou na sua escolha para a realização deste trabalho. Além disso, foi possível adequá-lo à estratégia da Dissertação e planejar a continuidade do desenvolvimento real do software para a Rede Biofoco, graças às suas principais características :

a) Orientado a caso de uso: esses casos foram utilizados como o principal artefato para o estabelecimento do comportamento desejado do Sistema, para a verificação e a validação da arquitetura, para a realização de testes e para a comunicação entre os participantes do projeto .

b) Centrado na arquitetura: a arquitetura está sendo utilizada como principal artefato para conceituação, construção, gerenciamento e evolução do Sistema, que ainda se encontra em desenvolvimento.

c) Iterativo e incremental: iterativo porque tem envolvido o gerenciamento de seqüências de versões prelimares executáveis, partindo das funcionalidades mais prioritárias, por exemplo “Visualizar alinhamento *pairwise*”. É incremental porque tem envolvido a integração contínua da arquitetura do Sistema para a produção dessas versões, de maneira que cada nova versão vem incorporando os aprimoramentos em relação às anteriores. Também está orientado a riscos, ou seja, cada nova versão tem como foco atacar e reduzir os riscos mais significativos para o sucesso do projeto. A cada fase, novos riscos podem ser detectados.

O Processo Unificado é um modelo bidimensional composto por: quatro fases e suas iterações; e ciclos de desenvolvimento e seus nove fluxos de trabalho (Figura 4.2-1). Uma fase é o intervalo de tempo decorrido entre dois importantes marcos do processo, quando um conjunto bem definido de objetivos é alcançado, os artefatos (documentos, relatórios ou programas executáveis) são concluídos e decisões são tomadas para se passar à fase seguinte. Uma iteração é uma subdivisão das fases e representa um ciclo completo de desenvolvimento, resultando em uma versão (interna ou externa) de um subconjunto do produto final em desenvolvimento, que cresce de modo incremental de uma iteração para outra até se tornar o sistema final. A passagem pelas quatro fases é

chamada de um ciclo de desenvolvimento, que se constitui de fluxos de trabalho, inclusive aqueles da metodologia tradicional de desenvolvimento em cascata.

Segue o esquema:

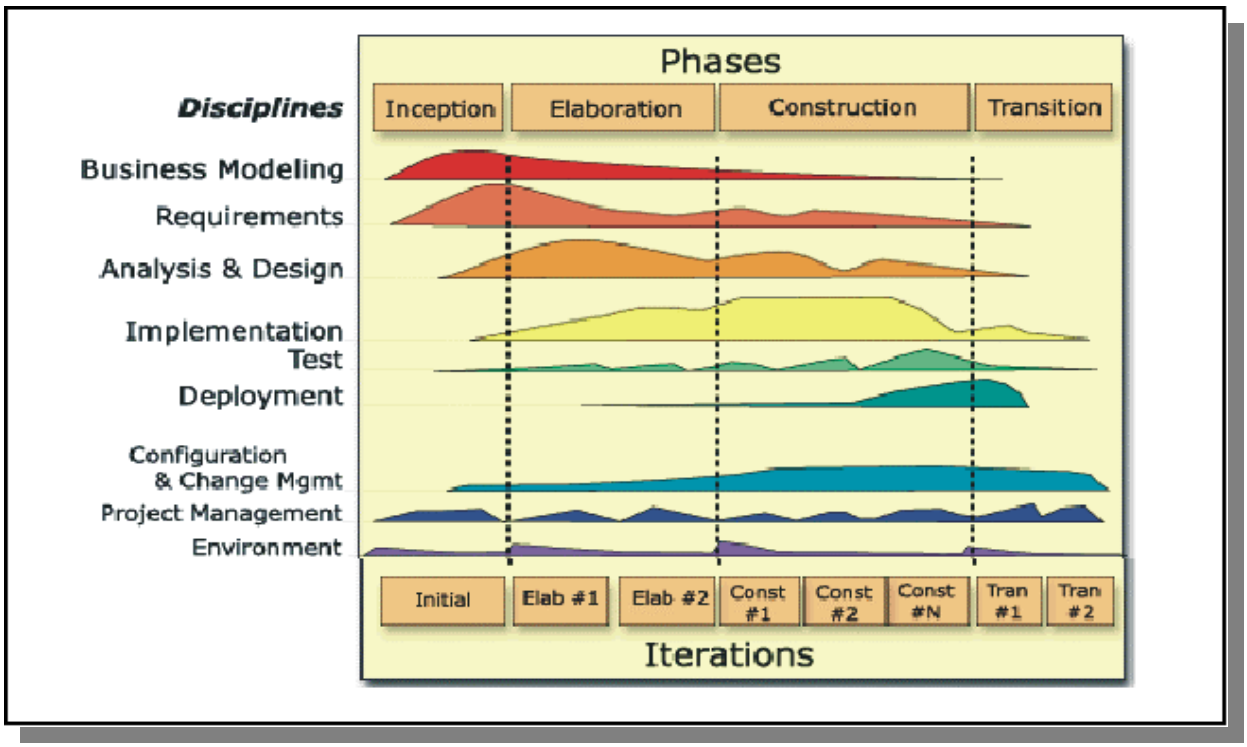

Figura 4.2-1 – Ciclo de vida do RUP.
Todas as palavras em inglês estão traduzidas no texto abaixo. Fonte: http://www.ambysoft.com/unifiedProcess.html.

De modo mais detalhado, as quatro fases podem assim ser descritas:

a) **Concepção** (ou *Inception*): é a primeira fase do processo. Para este trabalho de Dissertação, foram conseguidos os artefatos de : identificação do problema, identificação das necessidades do cliente, realização de estimativas superficiais, análise da viabilidade técnica e econômica do produto, proposta de um esboço para a arquitetura, elaboração do plano de desenvolvimento de software, detalhamento de alguns dos casos de uso mais significativos e planejamento da fase de elaboração.

b) **Elaboração** (ou *Elaboration*): é a segunda fase do processo. Para este trabalho de Dissertação, foram conseguidos os artefatos de : definição de uma arquitetura estável, detalhamento da maior parte dos casos de uso e apresentação de estimativas mais precisas para o projeto.

c) **Construção** (ou *Construction*): é a terceira fase do processo. Para este trabalho, estão sendo conseguidos os artefatos de : continuação do

detalhamento dos demais casos de uso e implementação dos casos de uso do Sistema.

d) **Transição** (ou *Transition*): é a quarta fase do processo, em que o software chega às mãos da comunidade de usuários. Para este trabalho, espera-se que sejam obtidos os artefatos de : configuração do ambiente, instalação, teste e homologação do Sistema em ambiente de produção.

Os nove fluxos de trabalho encontram-se subdivididos em dois grupos:

a) Fluxos de trabalho do processo:

- **Modelagem do negócio** (ou *business modeling*) : descreve a estrutura e a dinâmica da empresa. Para o Sistema desta Dissertação, este fluxo de trabalho está sendo executado conforme o Plano de Desenvolvimento.
- **Requisitos** (ou *requirements*): descreve o método baseado em casos de usos para identificar requisitos. Para o Sistema desta Dissertação, este fluxo de trabalho está sendo executado conforme o Plano de Desenvolvimento.
- **Análise e projeto** (ou *analysis and design*): descreve as várias visões da arquitetura. Para o Sistema desta Dissertação, este fluxo de trabalho está sendo executado conforme o Plano de Desenvolvimento.
- **Implementação** (ou *implementation*): leva em consideração o desenvolvimento do software, o teste da unidade e a integração. Para o Sistema desta Dissertação, este fluxo de trabalho está sendo executado conforme o Plano de Desenvolvimento.
- **Teste** (ou *test*): descreve casos de teste, procedimentos e medidas para acompanhamento de erros. Para o Sistema desta Dissertação, este fluxo de trabalho está sendo executado conforme o Plano de Desenvolvimento.
- **Entrega** (ou *deployment*): abrange a configuração do sistema a ser entregue. Para o Sistema desta Dissertação, este fluxo de trabalho está sendo executado conforme o Plano de Desenvolvimento.

b) Fluxos de trabalho de suporte:

- **Gerenciamento de alterações e configuração** (ou *deployment*): controla as modificações e mantém a integridade dos artefatos do projeto. Para o Sistema desta Dissertação, este fluxo de trabalho está sendo executado conforme o Plano de Desenvolvimento.

- **Gerenciamento de projeto** (ou *project management*): descreve várias estratégias para o trabalho com um processo iterativo. Para o Sistema desta Dissertação, este fluxo de trabalho está sendo executado conforme o Plano de Desenvolvimento.
- **Ambiente** (ou *environment*): abrange a infra-estrutura necessária para o desenvolvimento do sistema. Para o Sistema desta Dissertação, este fluxo de trabalho está sendo executado conforme o Plano de Desenvolvimento.

4.3 JAVA

É uma linguagem de programação de alto nível totalmente orientada a objetos, desenvolvida pela Sun Microsystems (http://java.sun.com) em 1995 e que já se tornou popular em todo o mundo . Inspirada no C++, Java foi projetado com a finalidade de ser obter um produto de pequenas dimensões, simples e portátil para diferentes plataformas e sistemas operacionais, seja em nível de código-fonte ou código binário (*bytecode*), ou seja, os programas podem ser executadas em qualquer computador em que esteja instalada a máquina virtual Java (*JVM – Java Virtual Machine*).

Java é atualmente utilizada para criar páginas da Web com conteúdo interativo e dinâmico, desenvolver aplicativos corporativos de larga escala, aprimorar funcionalidades de servidores Web e fornecer aplicativos para dispositivos destinados ao consumidor final (celulares, pagers, PDAs etc), sempre fazendo com que os componentes já desenvolvidos possam ser reutilizados.

Java traz, ainda, algumas características que facilitam o tratamento das questões inerentes a aplicações distribuídas, tais como: separação entre código e interfaces, tolerância a falhas através da manipulação de exceções, suporte a rede e objetos distribuídos, segurança e concorrência **[Pappas, 2002]**.

Esta plataforma - bem mais que uma simples linguagem de programação - está sendo utilizada para o desenvolvimento dos programas deste trabalho de Dissertação, numa interação muito estreita com a linguagem XML. A versão escolhida é a J2SDK1.4.1,

que já se encontra estável e traz embutida o Java Web Start, visando facilitar o acesso remoto à aplicação.

4.4 *DESIGN PATTERNS* E ORIENTAÇÃO A OBJETOS

[Sipert, 1998] Se se perguntasse para Engenheiros de Software por que a tecnologia de objetos é mais recomendada que as suas concorrentes, provavelmente a resposta mais freqüente seria "reuso". Na realidade, a reutilização de Componentes de software está diretamente relacionada a duas questões que afetam diretamente os projetistas e os usuários dos produtos: qualidade e produtividade.

É mais rápido montar um software a partir de Componentes predefinidos do que escrever todo o código desde o início. Além disso, o esforço necessário para se garantir confiabilidade e ergonomia[82] ao produto durante a fase de testes é consideravelmente menor quando o trabalho se fundamenta em elementos pré-fabricados. Por esse motivo, todos concordam que, atualmente, o ideal é desenvolver sistemas orientados a objetos. O objeto é uma unidade de software auto-gerenciável e com grande potencial para reuso. Note-se que é usada a expressão potencial para reuso, ou seja, o simples fato de usar tecnologia de objetos não garante isso. O reuso tem que ser planejado.

Nesse contexto, a essência dessas soluções já executadas pode ser apresentada como um padrão (*pattern*), que se repete em diversos problemas, os quais nem sempre se encontram relacionados (vide [xlix]). Por exemplo, de uma maneira grosseira, o diagnóstico e o tratamento de doenças na medicina podem ser comparados com a aplicação de padrões. Quando um paciente procura um médico, ele informa a sua situação atual e o médico tenta identificar sintomas de uma anomalia (problema). Ao identificar uma série de sintomas, o médico tenta encontrar os registros de outros casos semelhantes. Esses casos anteriores fazem parte da experiência do médico ou foram apresentados durante o seu curso de medicina. Ao detectar qual a moléstia do paciente, o médico tenta iniciar um tratamento (solução), que, na maioriada das vezes, vai apresentar efeitos benéficos (como a cura do paciente) e alguns efeitos colaterais não desejados (como desconforto durante o tratamento, imobilização etc). Esses efeitos em conjunto são

[xlix] BUSCHMANN, Frank; MEUNIER, Regine; ROHNERT, Hans; SOMMERLAND, Peter; STAL, Michael. **Pattern-Oriented *Software* Architecture: A System of Patterns.** John Wiley & Sons, 1996. 457p.

a conseqüência da aplicação da solução. Para facilitar a comunicação entre si, os médicos convencionaram nomes para as doenças **[Mangan, 1999]**.

Segundo Christopher Alexander[l], cada *pattern* descreve um problema que ocorre várias vezes em nosso ambiente e o núcleo de uma solução para esse problema, de forma que se possa reutilizá-la diversas vezes . Segundo a GoF[li], um *pattern*, em geral, é formado por quatro elementos essenciais, a saber **[Sipert, 1998]**:

a) Nome: é uma referência que pode ser usada para descrever um problema de projeto, suas soluções e conseqüências, através de uma ou duas palavras. A nomeação de um *pattern* aumenta o nosso vocabulário de projeto e nos permite trabalhar com um alto nível de abstração, pois podemos descrever uma solução inteira utilizando apenas uma expressão, ao invés de uma lista de objetos e seus relacionamentos.

b) Problema: descreve quando aplicar o *pattern*. Explica o problema e seu contexto. Algumas vezes, o problema incluirá uma lista de condições que devem ser satisfeitas antes de se aplicar o *pattern*.

c) Solução: descreve os elementos que compõem o projeto, seus relacionamentos, responsabilidades e colaborações. A solução não descreve implementação ou projeto concreto em particular, porque um *pattern* é como um molde que pode ser aplicado em várias situações diferentes.

d) Conseqüências: são os resultados, as vantagens e as desvantagens da aplicação do *pattern*. Geralmente envolvem questões de espaço, tempo, flexibilidade, extensiblidade, portabilidade, linguagem e implementação.

---

[l] Christopher Alexander - no final da década de 70, esse especialista passava por um problema relacionado à arquitetura e à engenharia civil. Percebendo que todas as construções de edifícios, que eram funcionais e confortáveis, possuíam algumas características em comum, resolveu catalogar essas soluções em seu livro *The Timeless Way of Building*. Durante anos, os especialistas na área de engenharia de software tentaram adequar as idéias de Alexander para a construção de sistemas de informação. Finalmente, em 1994, liderados por Erich Gamma, a GoF (Gang of Four) - Erich Gamma, Richard Helm, Ralph Johnson e John Vlissides publicaram o livro *Design Patterns: Elements of Reusable Object-Oriented Software*, que deu origem à onda dos Patterns na área de informática, pois é considerado o primeiro catálogo bem descrito sobre *patterns* de projeto para programas orientados a objeto. Após isto, seguiram outros livros de outros autores, publicações, papers, conferências, entre outros.

[li] GoF (Gang of Four) – Apelido dado ao grupo composto por Erich Gamma, Richard Helm, Ralph Johnson e John Vlissides. Publicaram o livro *Design Patterns: Elements of Reusable Object-Oriented Software*, que deu origem à onda dos *patterns* na área de informática.

Existem diferentes tipos de padrões, dentre os quais se destacam: padrões para projeto (*design patterns*[lii]), padrões para análise (*analysis patterns*[liii]) e padrões para organização de bancos de dados (*data patterns*[liv]). Existem também padrões propostos para outras áreas do conhecimento, como a arquitetura e urbanismo (vide [lv]) e administração (vide[lvi]) **[Mangan, 1999]**.

Uma observação pertinente é que as bibliotecas da linguagem Java fazem amplo uso de *design patterns*. Conseqüentemente, um bom conhecimento sobre esses padrões auxilia na compreensão dos recursos da linguagem e da programação orientada a objetos, de forma indireta.

Como os *Design patterns* resolvem problemas de projeto?

Alguns princípios básicos devem estar muito claros para aqueles que pretendem usar *design patterns* de forma apropriada. Em primeiro lugar, é importante entender a diferença entre a classe de um objeto e seu tipo. Uma classe define como um objeto é implementado, enquanto um tipo define apenas a interface de um objeto, ou seja, qual é o conjunto de requisições que ele é capaz de responder **[Sipert, 1998]**.

Também é importante entender a diferença entre herança[83] de classes e herança de interface (ou subtipo). Herança de classes define a implementação de um objeto com base na implementação de outro objeto, isto é, a herança de classes é um mecanismo para compartilhamento de código e representação. Por outro lado, herança de interface (ou subtipo) descreve quando um objeto pode ser usado no lugar de outro.

Um dos princípios de projeto reutilizável orientado a objetos é: programar orientado à interface, não à implementação. Uma conseqüência prática deste princípio é a recomendação de não se declararem variáveis como instâncias de classes concretas. Ao invés disso, devemos nos comprometer com o uso de interfaces definidas por classes

[lii] GAMA, Erich; HELM, Richard; JOHNSON, Ralph; VLISSIDES, John. *Design Patterns*: Elements of Reusable Object-Oriented *Software*. Addison-Wesley, 1995.
[liii] FOWLER, Martin. Analysis Patterns: Reusable Object Models. Addisson-Wesley, 1996.
[liv] Hay, David C. Data Model Patterns: Conventions of Thought.
[lv] ALEXANDER, Christopher; ISHIKAWA, Sara; SILVERSTEIN, Murray. A Pattern Language: Towns, Buildings, Construction. Oxford University Press, 1977. 1171p.
[lvi] DIKEL, David; HERMANSEN, Christy; KANE, David; MALVEAUX, Raphael. Organizational Patterns for *Software* Architecture. Proceedings... PLoP'97 Pattern Languages of Programming Conference. Disponível em http://st-www.cs.uiuc.edu/users/hanmer/PloP-97/Workshops.html .

abstratas. Devem-se usar *patterns* de criação para instanciar classes concretas. Isto significa que a associação entre uma interface e a sua implementação deve ocorrer de forma transparente, durante a instanciação.

Outro princípio é: usar preferencialmente composição de objetos em vez de herança de classes. Algumas desvantagens da herança são as seguintes: não se pode mudar as implementações herdadas das superclasses em tempo de execução; a herança expõe uma subclasse[84] aos detalhes de implementação da sua classe pai e isto quebra o encapsulamento[85]; a implementação de uma subclasse torna-se tão dependente da implementação da sua superclasse que qualquer mudança nesta forçará uma mudança naquela. Apesar disso, reutilização através de herança torna mais fácil a criação de novos componentes a partir de outros já existentes. Por essa razão, embora a recomendação seja priorizar o uso de composição, herança e composição são recursos complementares.

Design patterns não devem ser aplicados indiscriminadamente, pois, embora tornem os projetos flexíveis, podem também complicá-los e acrescentar-lhes um custo de performance. Um *design pattern* deve, portanto, ser aplicado somente quando a flexibilidade é realmente necessária. As seções de “conseqüências” são muito úteis para avaliar os benefícios e as desvantagens do *pattern*.

4.4.1 Os *patterns* mais utilizados no Sistema proposto

O *pattern* MVC

O padrão "Modelo-Visão-Controlador" (*MVC-Model-View-Controller*) consiste na definição de três classes distintas, onde o "View" representa a camada cliente ou de apresentação, o "Controller", a camada de lógica do negócio (gerência de eventos de interface e da aplicação), e o "Model", a camada de armazenamento ou de acesso aos dados **[Mangan, 1999]**.

Por exemplo, poder-se-iam combinar três classes para criar uma aplicação que permite a edição de um texto. A classe de apresentação de dados poderia ser um campo de texto (*textfield*), a classe de modelo poderia ser uma cadeia de caracteres (*string*) e o controlador cuidaria dos eventos provenientes da interface e da aplicação mantendo o valor da cadeia de caracteres e do campo texto consistentes. Então, se o usuário

apagasse o conteúdo da caixa de texto, o valor da cadeia de caracteres teria de ser alterado de acordo. Na maioria das linguagens de programação, a função do controller é desempenhada pelo próprio sistema geral de notificação de eventos. No caso de Java, o controller é implementado com base em *callbacks* (jdk 1.0) e *listeners* (jdk 1.1 em diante).

De uma forma geral, o MVC apresenta uma coleção de três classes que desempenham funções complementares e trabalham em cooperação entre si. Alguma experiência usando o MVC pode auxiliar em muito a habilidade de dividir uma aplicação em classes. Com a integração do sistema de eventos aos modernos sistemas operacionais e a evolução das linguagens de programação, o padrão MVC perdeu um pouco da sua utilidade original e muitas variações foram propostas a organização das classes.

Esse padrão se torna muito útil quando um mesmo dado (*model*) pode ser visualizado de formas diversas (*views*). Por exemplo, um conjunto de valores pode ser apresentado em uma planilha eletrônica como uma tabela ou como um gráfico. A tabela e o gráfico são duas apresentações para o mesmo modelo.

Para este trabalho de Dissertação, o *pattern* MVC está sendo utilizado principalmente para a estruturação do Serviço de Persistência, baseado no modelo POS (Persistent Object Service) do CORBA. Tanto o Serviço de Persistência quanto o modelo do CORBA serão explicados detalhadamente mais adiante.

O *pattern* Factory

O *pattern* Factory (ou fábrica) retorna uma instância de uma das diversas e possíveis classes, dependendo do parâmetro que lhe é passado. Usualmente, todas as classes que o método getClass( ) retorna possuem uma superclasse e métodos em comum, mas cada uma executa a tarefa diferentemente e é otimizada para diferentes tipos de dados **[Cooper, 2002]**. Segue a Figura 4.4-1:

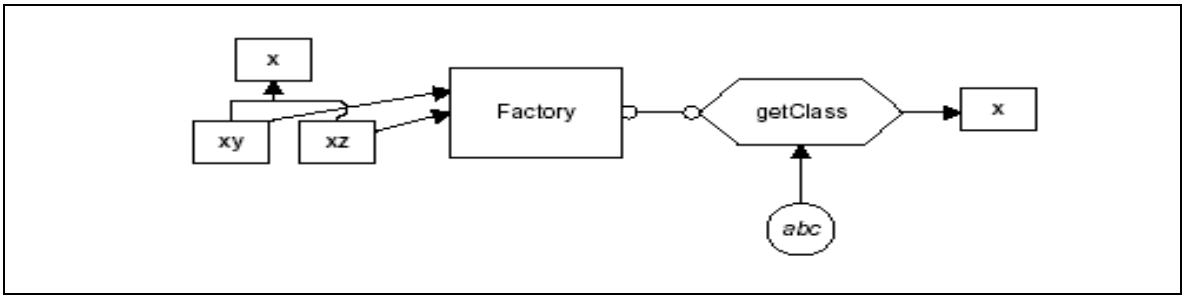

Figura 4.4-1 – Esquema do *pattern* Factory.
Fonte: **http://www.patterndepot.com/put/8/factory.pdf**.

Na Figura 4.4-1 mostrada acima, x é a superclasse e as classes xy e xz são suas subclasses. O *pattern* é a classe que decide qual dessas subclasses será retornada dependendo do parâmetro que lhe foi passado. Á a direita, está definido o método getClass( ) que recebe algum valor abc e retorna uma instância da classe x . De qualquer modo, não importa ao programador saber qual classe está sendo retornada, pois todas têm os mesmos métodos, mas implementações diferentes. Tudo é decidido pela fábrica, de um modo bastante simples.

O uso deste *pattern* deve ser considerado nas seguintes situações: uma classe não pose antecipar quais tipos de objeto ela deve criar; a classe usa sua subclasses para especificar quais objetos ela cria; e pode-se localizar a informação de quais classes foram criadas.

Existem diversas variações deste *pattern*: a) a superclasse é abstrata e o *pattern* deve retornar uma classe concreta; b) a superclasse contém métodos e apenas terá subclasses para os casos em que esses métodos sejam insuficientes; e c) os parâmetros são passados para fábrica para que esta saiba qual dos diversos tipos de classes deve ser retornado. Nesse caso, as classes podem compartilhar os mesmos nomes de métodos, embora esses sejam métodos polimórficos (vide polimorfismo[86]).

Para este trabalho de Dissertação, este *pattern* está sendo utilizado também para a estruturação das classes de controle do Serviço de Persistência, baseado no modelo POS (Persistent Object Service) do CORBA. Devido à sua importância, outros usos neste Sistema estão sendo estudados.

O Pattern Facade

Freqüentemente, à medida que os programas são desenvolvidos, cresce também a complexidade. Este *pattern* permite reduzir essa complexidade provendo uma interface simplificada que minimiza a comunicação e as dependências dos sub-sistemas, tornando-os mais reutilizáveis, portáveis, independentes e mais fáceis de serem mantidos. Também pode-se utilizar este *pattern* para dividir o sistema em camadas **[Cooper, 2002]**.

Um bom exemplo onde este *pattern* é utilizado e nas interfaces JDBC do próprio JAVA. Pode-se conectar a qualquer banco de dados para o qual o fornecedor tenha uma implementação concreta dessas interfaces.

Segue a Figura 4.4-2:

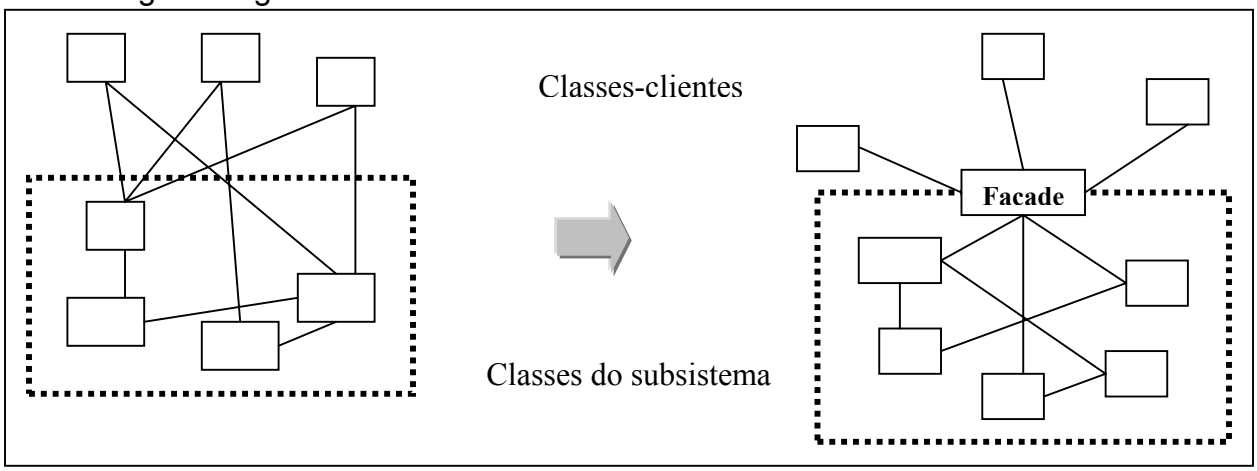

Figura 4.4-2 – Esquema do *pattern* Facade.
Baseado no livro *Design Patterns – Elements of Reusable Object-Oriented Software*, Gama et al, página 185.

Para este trabalho de Dissertação, este *pattern* está sendo utilizado para a estruturação de todas as interações com as ferramentas externas. Há classes persistentes que mantêm os parâmetros e o *pattern* os utiliza para execução da ferramenta apropriada, controle dos arquivos de retorno etc.

4.5 CORBA

CORBA (***C**ommon **O**bject **R**equest **B**roker **A**rchitecture* ou Arquitetura Comum de Agente de Requisição de Objeto) consiste em uma definição aberta e neutra que especifica arquitetura e infra-estrutura para a execução de programas em redes de computadores. A sua grande vantagem é permitir a integração de programas escritos em

diversas linguagens e executados em virtualmente qualquer combinação de hardware ou sistema operacional. Ao mesmo tempo, provê uma sólida base para a utilização de componentes reutilizáveis de software. Consulte http://www.omg.org.

Dentre os diversos serviços do CORBA, três deles estão sendo diretamente utilizados no Sistema BLOOM: Serviço de Objeto Persistente, Serviço de Consulta e Serviço de Coleção.

O Serviço de Objeto Persistente (*POS – Persistent Object Service*) permite que objetos sejam persistidos além da aplicação que o cria ou do cliente que o usa. O ciclo de vida pode ser curto ou indefinido. Este serviço permite que o estado de um objeto seja salvo em um meio persistente e que esse estado seja recuperado sempre que necessário. Quando o objeto está na memória local, pode-se acessar seus atributos e métodos à velocidade das linguagens de programação nativas **[Orfali et al, 1997]**.

Os principais elementos deste serviço são:

a) **Objetos persistentes (PO´s)**: são objetos cujo estado deve ser persistido ou armazenado. Para tornar-se também persistente, um objeto deve herdar as características de um PO. Todo objeto deste tipo tem um PID (Persistent Identifier). É descrito pela interface PO.
b) **Gerenciamento de Objeto Persistente (POM)**: é uma interface independente de implementação para operações persistentes. Posiciona-se entre os objetos e o banco de dados.
c) **Serviço de Dados Persistentes (PDS´s)**: são interfaces para implementações particulares de um meio de armazenado (ou banco de dados). Executa a tarefa de mover dados de um objeto para o repositório persistente. Para conseguir acesso ao meio de armazenamento, é necessário especializar as interfaces PDS e PID.
d) **Meio de armazenamento**: são repositório de dados persistentes para os atributos de um objeto. São exemplos os ODBMS, RDBMS, dentre outros.

Além das interfaces citados nos elementos acima, há três outras também muito importantes neste serviço: a) PIDFactory, que permite criar um objeto PID; b)POFactory,

que permite criar uma instância de um objeto persistente; e c) PID, que retorna uma versão string do PID.

Nesta Dissertação, este Serviço de Objeto Persistente estará representado para o Sistema BLOOM no item Mecanismos de Persistência.

O Serviço de Consulta (*OQS –Object Query Service*) permite encontrar objetos cujos atributos coincidam com o(s) critério(s) especificado(s) na busca. Deve-se notar que as consultas não possuem acesso ao estado interno do objeto, o que significa que não violam o encapsulamento. As consultas podem ser formuladas em OQL (Object Query Language), SQL (Structure Query Language), ou em subconjunto dessas duas linguagens **[Orfali et al, 1997]**.

O Serviço de Coleção (*CCS – CORBA Collection Service*) provê uma maneira uniforme para criar e manipular as maioria das coleções mais comuns. As classes deste serviço relacionam-se com as do Serviço de Consulta.

Coleções permitem que os objetos sejam manipulados em grupo. Tipicamente se aplicam operações de coleção em grupos, e não em objetos individuais. Exemplos de coleção são filas, pilhas, listas, arrays, árvores, conjuntos e bags, cada um exibindo comportamentos específicos do tipo de coleção. Alguns operações que podem ser invocadas em um set são: adicionar um novo membro, testar igualdade, testar se está vazio, união, intersecção, e assim por diante. Algumas coleções são ordenadas; outras usam chaves para identificar os elementos. Finalmente, algumas coleções rastream as referências dos objetos; outras simplesmente coletam tipos de dados (por exemplo, linhas em uma coleção do tipo tabela).

As principais interfaces de Coleção e de Consulta são:

a) **CollectionFactory**: define uma única operação create( ). Esse método deve ser invocado para criação uma nova instância de uma coleção vazia.

b) **Collection**: define operações que permitem adicionar, alterar, recuperar e remover membros de uma coleção. Invoca-se o método add_all_elements para adicionar todos os elementos de uma outra coleção para a coleção de destino. Insert_element_at permite adicionar

um elemento numa posição particular. Create_iterator permite criar um ponteiro móvel para navegar pela coleção.

c) **Iterator**: define três operações que permitem percorrer uma coleção. Reset aponta para o início da coleção. Next incrementa a posição do iterator. More permite testar se ainda restam elementos na iteração.

d) **QueryEvaluator**: define a operação evaluate( ) para avaliar uma consulta. Essa operação utiliza a linguagem de consulta especificada ou default. Um banco de dados é um exemplo de um objeto QueryEvaluator, pois gerencia uma coleção de objetos persistentes.

e) **QueryManager**: é uma forma mais poderosa de um QueryEvaluator. Também permite criar um objeto Query.

f) **Query**: define quatro operações que podem ser executadas em uma instância de Query. Prepare permite compilar a consulta e prepará-la para execução. Execute permite executar uma query compilada. Get_status permite determinar o status de preparação/execução da query. Get_result permite obter o resultado da query.

g) **QueryableCollection**: não introduz novas operações. Ao contrário, herda suas funcionalidades de QueryEvaluator e Collection. Objetos desta classe avaliam uma query em membros particulares de uma coleção. Deve-se notar que qualquer membro da coleção a ser avaliada por ser um objeto QueryableCollection. Isso significa que se pode ter um infinito número de subconsultas aninhadas.

Nesta Dissertação, os Serviços de Consulta e de Coleção estarão representados nos diagramas de seqüências e de classes dos Sistemas BLOOM e Genoma.

Utilizando a arquitetura CORBA e os serviços descritos acima, está sendo possível tratar os resultados de seqüenciamento como objetos que são distribuídos através da rede para o servidor central. Estes estão sendo atualmente depositados em bancos de dados relacionais administrados pelo programa gratuito PostgreSQL (http://www.postgresql.org), caso respeitem diversos critérios de qualidade **[Pappas, 2002]**.

Um sistema, conforme o que está sendo proposto, encaixa-se perfeitamente nessa filosofia, pois os laboratórios em rede podem depositar dados genômicos e receber

análises de seqüências. Implementações recentes demonstram a importância desta nova tecnologia na bioinformática (http://corba.ebi.ac.uk/), principalmente para a integração de bancos de dados biológicos **[Pappas, 2002].**

4.6 XML

Um documento XML (e***X****tended* ***M****arkup* ***L****anguage* ou Linguagem Extensível de Marcações) é uma unidade de informação que pode ser vista de dois modos: como uma seqüência linear de caracteres que contém dados do tipo caracter e marcações (tags), ou como uma estrutura abstrata que uma árvore composta de nós rotulados e decorados com atributos. Os dados vêm com uma quantidade enorme de informações a respeito de si mesmos (metadados) **[Allamaraju, 2000]**.

Lançada em dezembro de 1997, a linguagem XML representa um novo padrão para a organização de documentos (de forma estruturada e hierárquica), troca de dados entre sistemas (mesmo incompatíveis), compartilhamento de dados interplataforma, armazenamento de dados em arquivos ou bancos de dados. Mais informações, consulte http://www.xml.org.

Em relação ao HTML (*HyperText Markup Language*), que é o padrão de documentos da web, XML é um avanço e uma complementação àquele, mas não seu substituto. XML foi projetado para descrever os dados e focalizar nos seus significados. Já o HTML, este foi projetado para formatar e mostrar dados. Diferentemente de HTML, XML não possui tags pré-definidas e não é executável, ou seja, o usuário deve definir suas próprias tags e determinar o software que poderá enviar, receber, mostrar ou armazenar o arquivo no seu formato original ou o conteúdo dessas tags.

XML utiliza um DTD (*Document Type Definition ou Definição de Tipo de Documento*) ou um XML Schema (*Esquema XML*) para descrever os dados e manter suas regras de validação. Os esquemas são projetados para serem auto-descritivos. Vários autores apontam vantagens no uso do XML Schema, pois este é totalmente XML, permite a definição de tipos de dados próprios, utilizada descrições mais detalhadas, é mais modular e mais fácil de reusar. Utilizando esses esquemas, ferramentas como o Castor (http://castor.exolab.org) e o JAXB (http://java.sun.com/xml/jaxb/) convertem o documento

XML em objetos JAVA (*unmarshaling*), ou vice-versa (*marshaling*) **[Brodkin, 2001]**. Vide Figura 4.6-1 a seguir:

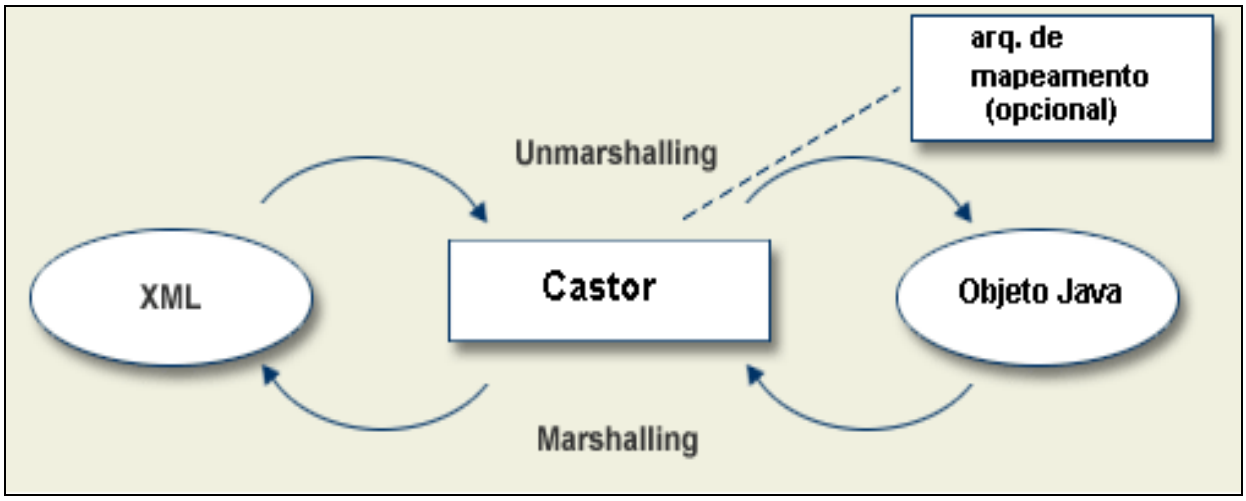

Figura 4.6-1 – Fluxo de dados na ferramenta Castor.
Fonte: **http://builder.com.com/article.jhtml?id=u00320021021HXO01.htm&vf=tt**.

No Sistema proposto nesta Dissertação, o XML funcionará como "uma moeda de troca entre os diversos tipos de saídas de programas de bioinformática". A partir dos dados armazenados, o processo automático de análise é iniciado. A utilização do CORBA traz benefícios adicionais, pois permite integrar os diversos programas de análise de seqüência viabilizando a sua execução nos moldes de uma linha de montagem. Estes programas encontram-se disponíveis, mas são codificados em diversas linguagens diferentes. A proposta é criar adaptadores universais (utilizando o *pattern* facade) para a chamada de tais programas, de modo que eles se associem como módulos independentes ao sistema geral, permitindo que sua execução seja distribuída, parametrizada e, ao mesmo tempo, facilitando a adição de novos módulos. Os resultados desses programas serão traduzidos para XML permitindo uma melhor integração e garantindo uma maior consistência nas análises em geral **[Pappas, 2002]**.

O XML já vem sendo utilizado no Sistema Genoma para *unmarshaling* e *marshaling* dos resultados obtidos do programa BLAST (vide Anexo). No Sistema BLOOM, o uso de XML será expandido para tratar o retorno de cada ferramenta executada e para realizar a persistência dos atributos das classes de parametrização e de filtragem de resultados.

5. VISÃO GERAL DO PROJETO

5.1 PROPRIEDADE INTELECTUAL

A propriedade intelectual no caso do software desenvolvido será restrita aos seus desenvolvedores e às instituições às quais esses são filiados. A utilização dos serviços será totalmente aberta e os programas serão distribuídos para todas as instituições participantes da Biofoco **[Pappas, 2002]**.

As co-autorias serão definidas de acordo com princípios de ética da ciência: a obrigação de inclusão do nome de um colega como co-autor de trabalho é definida pela sua participação intelectual na criação do conhecimento, conforme as normas universalmente aceitas pelos órgãos governamentais e pelo Ministério da Ciência e Tecnologia.

5.2 PLANO DE DESENVOLVIMENTO

Fase	Início	Fim	Objetivos	Critério de aceitação
Concepção	01/09/02	31/10/02	- Identificação do problema do usuário. - Identificação das necessidades do usuário. - Realização de estimativas superficiais. - Análise da viabilidade técnica e econômica do produto. - Proposição de um esboço para a arquitetura. - Elaboração do plano de desenvolvimento de software. - Detalhamento dos casos de uso mais significativos. - Planejamento da fase de Elaboração.	- Concordância entre clientes e equipe de desenvolvimento sobre escopo e estimativas superficiais do projeto.
Elaboração	01/11/02	20/02/03	- Definição de uma arquitetura estável. - Detalhamento da maior parte dos casos de uso. - Apresentação de estimativas mais precisas para o projeto. - Planejamento da fase de Construção.	- A arquitetura deve estar validada e implementada. - Concordância entre os interessados sobre prazos e custos.
Construção	21/02/03	04/04/04	- Continuação do detalhamento dos demais casos de uso. - Implementação dos casos de uso. - Planejamento da fase de Transição.	- O produto deve estar pronto para ser disponibilizado.
Transição	05/04/04	30/04/04	- Disponibilização do produto.	- Satisfação do cliente em relação ao produto.

Tabela 5.2-1 – Plano de desenvolvimento.

5.3 OPORTUNIDADE CIENTÍFICA

Este sistema é uma iniciativa dos pesquisadores da Rede Biofoco, no auxílio à execução integrada das diversas ferramentas em suas atividades em projetos genômicos. Inicialmente estão previstas as seguintes atividades: construção e visualização dos alinhamentos *pairwise* e múltiplo; predição e visualização de estrutura secundária de proteínas; e derivação e visualização de árvore filogenética.

Embora existam ferramentas que separadamente realizem as atividades propostas, alguns dos ganhos desta oportunidade científica são: redução de tempo para executar as tarefas; padronização; repositório único dos dados; integração das ferramentas, possibilitando fazer com que os dados de saída de uma sejam entrada para a outra (exemplo: seqüências do alinhamento *pairwise* são selecionadas como entrada para o alinhamento múltiplo); automatização de algumas tarefas, por exemplo a reexecução de consulta BLAST; e automatização de consultas no BD *pairwise* XML, conforme a lista de preferências do pesquisador.

À medida que um componente do Sistema esteja devidamente testado e aprovado pelos pesquisadores da Rede Biofoco, esse já será disponibilizado como serviço para os demais usuários externos à Rede, para que possam utilizá-lo, testá-lo e propor melhorias.

5.4 DETALHAMENTO OU SENTENÇA DO PROBLEMA

1. O problema	Escassez de produtos de livre acesso para manipulação de informações e análise elementar nos primeiros estágios de maturação de dados de seqüências em projetos de mapeamento genômico.
Afeta (quem)	Todos os pesquisadores interessados em pesquisa na área de bioinformática, inclusive os integrantes da Biofoco.
O impacto disto é	Análise incompleta ou trabalhosa das seqüências de um determinado projeto.
Uma solução de sucesso permitirá	Facilidade nas análises e melhor aproveitamento dos resultados do BLAST.
2. O problema	As versões atuais das ferramentas de livre acesso estão desatualizadas quanto à tecnologia, apresentam deficiências informacionais, possuem poucos recursos de

	interatividade e de acesso aos dados. Exemplo: WebBlast.
Afeta (quem)	Todos os pesquisadores interessados em pesquisa na área de bioinformática, inclusive os integrantes da Biofoco.
O impacto disto é	Alguns pesquisadores tentam criar soluções próprias ou adaptar as já existentes e incompletas. Outros, no entanto, devido aos problemas, acabam nem utilizando essas ferramentas de apoio. Exemplo: adaptação do WebBlast para o Sistema Genoma.
Uma solução de sucesso permitirá	Uma maneira única e atualizada de tratar os dados resultantes de uma consulta BLAST.
3. O problema	Impossibilidade de filtrar a grande quantidade de seqüências similares obtidas (*hits*) após a execução do BLAST, dado um conjunto de critérios do tipo pontuação, E-value, P-value, qualidade PHRED etc.
	Todos os pesquisadores interessados em pesquisa na área de bioinformática, inclusive os integrantes da Biofoco.
O impacto disto é	Muitos pesquisadores tendem a considerar apenas as primeiras ocorrências da lista, o que pode ser um equívoco bastante grave, pois a ordem em que os registros são apresentados não garante sua importância para um domínio específico.
Uma solução de sucesso permitirá	Filtragem dos resultados conforme um conjunto de critérios específicos para reduzir a quantidade de seqüências a serem analisadas.
4. O problema	Inexistência de ferramenta de análise dos resultados BLAST armazenados no BD *pairwise* XML, que integre: anotações das seqüências envolvidas; construção e visualização de alinhamento múltiplo; predição de estrutura secundária de proteína; derivação de árvore filogenética etc.
Afeta (quem)	Todos os pesquisadores integrantes da Biofoco.
O impacto disto é	Até mesmo para atingir propósitos idênticos dentro de um projeto de mapeamento genômico, pesquisadores vêm usando ferramentas diversas e não padronizadas. Exemplo: as diversas ferramentas disponíveis na Internet para realizar as análises mencionadas acima.
Uma solução de sucesso permitirá	Uma maneira única e integrada de tratar os dados armazenados no BD *pairwise* XML. Com isso, a potencialidade desses dados será ainda mais valorizada.
5. O problema	Impossibilidade - quanto aos produtos atuais de livre acesso - de juntar seqüências externas aos resultados do BLAST para se executar um alinhamento múltiplo.

Afeta (quem)	Todos os pesquisadores interessados em pesquisa na área de bioinformática, inclusive os integrantes da Biofoco.
	A execução do alinhamento múltiplo sempre fica restrita à lista de seqüências recuperadas na consulta BLAST, sem que haja possibilidade de se agregarem sequências de outros projetos e fazer um estudo comparativo.
Uma solução de sucesso permitirá	Expandir a lista de seqüências a serem submetidas a alinhamento múltiplo.
6. O problema	Impossibilidade - quanto aos produtos atuais de livre acesso - de criação e manutenção de uma lista de preferências (*wish list*), para que o pesquisador possa ser automaticamente avisado por e-mail de que novas informações de seu interesse estão armazenadas no BD *pairwise* XML.
Afeta (quem)	Todos os pesquisadores integrantes da Biofoco.
O impacto disto é	Os pesquisadores precisam realizar manualmente a busca por novas informações relativas às seqüências de um projeto. Em alguns casos, isso significa reexecutar uma consulta BLAST.
Uma solução de sucesso permitirá	Economia de tempo e ganho de performance no conhecimento das novas informações disponíveis sobre os assuntos assinalados na lista de preferências.
7. O problema	Impossibilidade - quanto aos produtos atuais de livre acesso - de agendar e reexecutar atividades relativas às seqüências de interesse armazenadas no BD *pairwise* XML, por exemplo uma nova consulta BLAST.
Afeta (quem)	Todos os pesquisadores integrantes da Biofoco.
	Os dados armazenados tornam-se obsoletos (desatualizados) em um curto período de tempo, o que requer atualizações freqüentes. Pode ser algo trabalhoso e que nem todo pesquisador dispõe de tempo para efetuar.
	Economia de tempo e de esforço para execução e reexecução de atividades, evitando esquecimentos e, principalmente, obsolescência de dados.
8. O problema	Inexistência de uma ferramenta orientada a objetos que atenda aos propósitos listados acima, que permita reuso e inserção gradual de novas funcionalidades, utilizando objetos ou componentes distribuídos de maneira segura.
Afeta (quem)	Todos os pesquisadores interessados em pesquisa na área de bioinformática, inclusive os integrantes da Biofoco.

O impacto disto é	As arquiteturas das ferramentas atuais não permitem: expansão gradual do aplicativo, reuso de código, segurança do servidor de dados e integridade das informações que trafegam na Internet.
Uma solução de sucesso permitirá	Utilizando metodologias atuais como UML e RUP, um aplicativo robusto poderá ser construído com tecnologias do tipo JAVA, XML, J2EE, Corba etc.

Tabela 5.4-1 – Detalhamento ou sentença do problema.

5.5 POSICIONAMENTO DO PRODUTO

Para	Instituições ou pesquisadores das áreas de biologia, bioinformática e demais áreas afins.
Que	Desejam reduzir tempo e esforço para executar as diversas ferramentas de apoio em bioinformática.
O Sistema BLOOM	É um aplicativo para execução da ferramenta desejada e visualização gráfica do resultado.
Que	Possibilita a integração das diversas ferramentas de bioinformática disponíveis na Internet, fazendo com que os dados de saída de uma sejam utilizados como entrada para a ferramenta a ser chamada em seguida.
Diferentemente	De algumas soluções de acesso livre já disponíveis na Internet, que executam com propriedade uma ou outra ferramenta, mas não se dispõem a uma integração do porte previsto neste trabalho.
Nosso produto	Será de acesso livre e código aberto. A implementação ocorrerá conforme a disponibilidade de programadores da Rede Biofoco.

Tabela 5.5-1 – Posicionamento do produto.

5.6 DESCRIÇÃO DA EQUIPE CLIENTE E DOS USUÁRIOS

Equipe Cliente	Descrição	Responsabilidade
Equipe de pesquisadores do Laboratório de Bioinformática da Embrapa/Recursos Genéticos.	São Mestres e Doutores nas áreas de biologia e bioinformática lotados no Laboratório dessa unidade da Embrapa.	-Definir o escopo do Sistema; -Aprovar os requisitos levantados e os modelos gerados; -Acompanhar o andamento do projeto.
Equipe de pesquisadores do Laboratório de Bioinformática da UCB.	São Mestres e Doutores nas áreas de biologia e bioinformática lotados no Laboratório de Bioinformática da UCB.	-idem.
Equipe de pesquisadores do Laboratório de Bioinformática da UNB.	São Mestres e Doutores nas áreas de biologia e bioinformática lotados no Laboratório de Bioinformática da UNB.	-idem.

Tabela 5.6-1 – Equipe cliente.

Usuários
Pesquisadores da área de bioinformática
Pesquisadores da área de biologia molecular
Estudantes de biologia e de bioquímica
Pós-graduandos nas áreas afins
Mestrandos nas áreas afins
Doutores nas áreas afins
Professores universitários nas áreas afins
Empresas de biotecnologia
Indústrias de produção de alimentos (OGM)
Empresas de produção animal (OGM)
Orgãos fiscalizadores do governo e sociedade
ONG´s (Organizações Não-Governamentais) nas áreas afins

Tabela 5.6-2 – Potenciais usuários do Sistema.

5.7 INTERAÇÕES OU PERSPECTIVAS DO PRODUTO

O Sistema, por si só, não realiza todas as atividades do processo; ao contrário, está planejado para, sempre que necessário, interagir com outros aplicativos e obter desses um serviço, a saber:

a) **Sistema Genoma**: é um sistema de anotação genômica produzido no Laboratório de Bioinformática da EMBRAPA Recursos Genéticos e Biotecnologia. O sistema integra programas de análise e visualização de seqüências nucleotídicas e cromatogramas originais produzidos em equipamentos de seqüenciamento automático (Megabace, ABI 3700, ABI 3100). Iniciando o processo, o arquivo obtido do seqüenciador (o cromatograma) é submetido ao programa PHRED para se obterem o arquivo no formato FASTA e o arquivo de qualidade da seqüência. Em seguida, executa-se o Crossmatch contra o banco de vetor para eliminação de seqüências de vetores. Após isso, as seqüências de baixa qualidade são desprezadas e apenas aquelas com tamanho mínimo e qualidade aceitáveis são armazenadas, para as quais são feitas totalizações de clones na placa, clones na biblioteca etc. Para finalizar , executa-se o BLAST para cada uma das novas seqüências contra os bancos selecionados para aquele projeto.

A segurança dos bancos de dados gerados é garantida pelo sistema utilizando contas e senhas próprias para cada usuário.

O sistema GENOMA foi elaborado com a vantagem da linguagem JAVA e sua portabilidade, que permite a execução do programa em sistemas operacionais LINUX, WINDOWS e MAC. Além dessa vantagem, o Sistema foi montado sob a arquitetura CORBA, que define uma plataforma de orientação de objetos com uma linguagem para descrição de interfaces (IDL) com mapeamentos padronizados em diversas linguagens para um conjunto de serviços básicos. Devido ao seu porte e à metodologia utilizada, o Sistema ainda se encontra em desenvolvimento.

É este Sistema que obtém o resultado do BLAST no formato XML, faz o *marshaling* e o *unmarshaling* das tags resultantes e mantém os dados no BD *pairwise* XML. Após logar-se no Sistema Genoma, o Pesquisador escolhe a opção de Consulta, Análise de Seqüência, Resultados do BLAST e inicia o acesso ao Sistema BLOOM.

b) **BD *pairwise* XML**: é o banco de dados que armazena resultados de alinhamento *pairwise* obtidos pelo uso da ferramenta BLAST, no formato XML. Há também outras

tabelas auxiliares. A obtenção e o armazenamento prévios dos dados de alinhamento no BD *pairwise* XML são efetuados pelo Sistema Genoma.

c) **Módulo de anotação**: possibilita obter os dados de anotação de uma determinada seqüência genômica. Esses dados ficam armazenados nos principais bancos de dados de seqüências disponíveis na Internet.

O Sistema automaticamente chamará este Módulo para obter a anotação de cada seqüência listada no resultado do alinhamento *pairwise*.

d) **Aplicativo de alinhamento múltiplo**: é uma ferramenta de largo uso e disponível na Internet para alinhamento de três ou mais seqüências de nucleotídeos ou aminoácidos, em busca de similaridade ou homologia entre essas. Não é necessário realizar nenhuma busca em bancos de dados de seqüências. Cada seqüência envolvida é comparada com todas as demais, isto é, comparação de todas contra todas.

A partir da apresentação gráfica do resultado do alinhamento *pairwise*, o pesquisador selecionará as seqüências que serão utilizadas como parâmetro de entrada para este aplicativo.

Para tornar a execução mais abrangente, será apresentada uma lista das ferramentas apropriadas, sendo que inicialmente será utilizado apenas o aplicativo ClustalW. Qualquer que seja a ferramenta escolhida pelo pesquisador, a adequação aos formatos de entrada e saída será tratada pelo próprio Sistema, utilizando o *pattern Façade*.

e) **Editor de alinhamento múltiplo**: é uma ferramenta de largo uso e disponível na Internet para manipular graficamente o resultado do alinhamento múltiplo.

Para tornar a execução mais abrangente, será apresentada uma lista das ferramentas apropriadas, sendo que inicialmente será utilizado apenas o aplicativo JalView. Qualquer que seja a ferramenta escolhida pelo pesquisador, a adequação

aos formatos de entrada e saída será tratada pelo próprio Sistema, utilizando o *pattern Façade*.

f) **Aplicativo de predição de estrutura secundária de proteína:** é uma ferramenta de largo uso e disponível na Internet para predizer a estrutura secundária de uma proteína (o início do enovelamento em 3D), a partir de uma seqüência genômica. Conhecer a estrutura é pré-requisito para um entendimento completo da função que essa proteína desempenhará. Esse entendimento tem aplicações potenciais em vários projetos genômicos, tais como mapeamento de funções de proteínas em vias metabólicas[87] *(metabolic pathways)* para genomas completos e dedução de relações evolutivas.

A partir da apresentação gráfica do resultado do alinhamento múltiplo, o pesquisador selecionará as seqüências que serão utilizadas como parâmetro de entrada para este aplicativo.

Para tornar a execução mais abrangente, será apresentada uma lista das ferramentas apropriadas. Qualquer que seja a ferramenta escolhida pelo pesquisador, a adequação aos formatos de entrada e saída será tratada pelo próprio Sistema, utilizando o *pattern Façade*.

g) **Aplicativo de árvore filogenética**: é uma ferramenta de largo uso e disponível na Internet para reconstruir a correta genealogia[88] entre organismos e estimar o tempo transcorrido desde que eles se divergiram de um ancestral comum. As relações evolutivas entre grupos de organismos são ilustradas em gráficos chamados de árvores filogenéticas.

A partir da apresentação gráfica do resultado do alinhamento múltiplo, o pesquisador selecionará as seqüências que serão utilizadas como parâmetro de entrada para este aplicativo.

Para tornar a execução mais abrangente, será apresentada uma lista das ferramentas apropriadas, sendo que inicialmente será utilizado apenas o aplicativo

Protpars. Qualquer que seja a ferramenta escolhida pelo pesquisador, a adequação aos formatos de entrada e saída será tratada pelo próprio Sistema, utilizando o *pattern Façade*.

Segue a Figura 5.7-1:

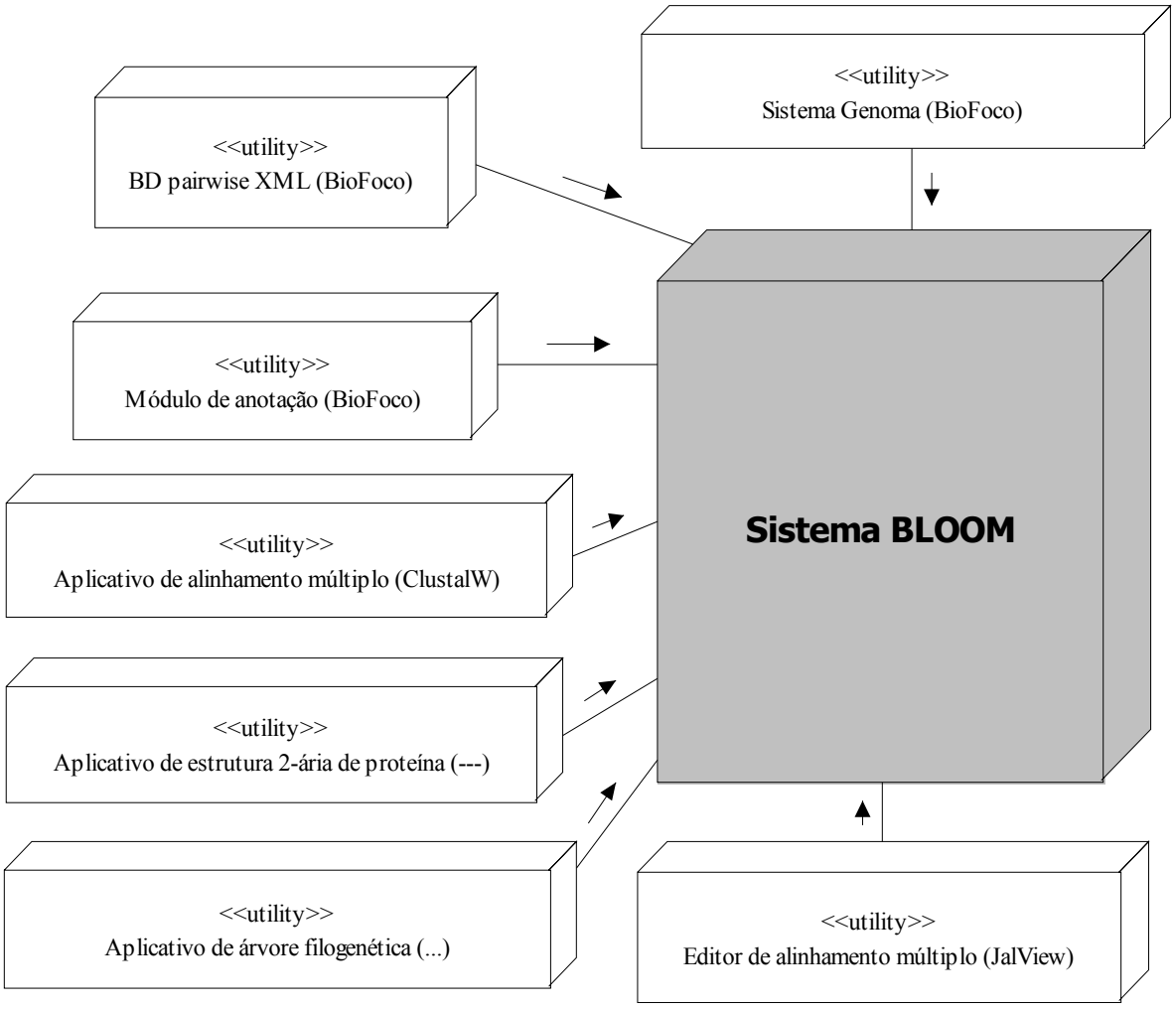

Figura 5.7-1 – Esquema das interações do Sistema.

5.8 APLICABILIDADE DE PADRÕES

Para garantir que o aplicativo possa ser efetivamente utilizado por um grande número de pesquisadores e que seja flexível o bastante para se adaptar às diversas situações dos projetos genomas, alguns padrões serão aplicados:

a) Software aberto: o sistema deverá seguir padrões de software aberto, inicialmente para sistema operacional (Linux), servidor HTTP (Apache) e servidor de banco de dados (PosgreSQL), barateando os custos com licenças.

b) Portabilidade: o programa deve ser executável em qualquer plataforma independente do tipo de máquina, sistema operacional e servidor de banco de dados.

c) Interconectividade: capacidade de acessar e distribuir dados através da Internet. Esta propriedade é fundamental, visto que os bancos de dados biológicos são muito volumosos, o que dificulta a manutenção local destes. Sendo assim, deve-se ser capaz de acessar esses repositórios de dados remotamente.

d) Modularidade: é um dos principais objetivos. Consiste na capacidade de o programa receber a adição de módulos independentes, que acrescentarão algum tipo de funcionalidade. Esses módulos poderão ser criados por outros grupos e integrados transparentemente ao programa. Com essa filosofia de criação, pretende-se enriquecer o espectro de serviços do programa através da descentralização do desenvolvimento. É importante também salientar que se deve estabelecer uma estrutura interna de compartilhamento de dados e resultados de forma a realmente se integrar os módulos e permitir que o programa tenha controle efetivo sobre o processamento.

e) Interface gráfica: presença de uma rica interface gráfica de interação com o usuário, visando permitir o acesso simples e rápido das funcionalidades, bem como fornecer os resultados de forma intuitiva para facilitar o seu entendimento.

f) Uso da API Java 2D para implementar os visualizadores gráficos.

g) Uso do padrão CORBA para comunicação entre os objetos.

h) O sistema deverá ser modelado utilizando Linguagem Unificada de Modelagem (UML), padrão da Object Management Group.

i) Uso do Processo Unificado Rational (RUP) para acompanhamento do processo de desenvolvimento do Sistema.

j) Integração com padrões da OMG para lifescience (XML).

k) A linguagem de desenvolvimento deverá ser Java, seguindo as especificações da Sun, para permitir maior interoperabilidade.

l) O sistema deverá ser disponibilizado em português, inglês e espanhol.

5.9 LISTA DE RISCOS

Risco	Impacto	Situação	Materialização	Plano de contingência
Prazo de desenvolvimento	Alto	Pendente	As principais funcionalidades do Sistema BLOOM precisam estar em produção no prazo estabelecido pelas três unidades participantes.	Após a fase de elaboração, quando houver mais subsídios para a realização de estimativas, deverá ser feita uma análise sobre a viabilidade dos custos e prazos do projeto.
Utilização do SGBD PosgreSQL	Alto	Pendente	Quando o sistema for posto em produção, existe a possibilidade de que o SGBD, sugerido para o desenvolvimento, não suporte os requisitos de volume de transação e desempenho.	Caso o PosgreSQL não apresente a disponibilidade desejada, podem ser utilizadas três alternativas para contornar o problema: 1)Contratação de especialistas no banco de dados para buscar uma otimização do serviço. 2)Treinamento, rápido, de membros da equipe de desenvolvimento para administração do SGBD. 3)Identificar uma alternativa de SGBD que apresente baixo custo e suporte os requisitos da aplicação.
Integração com aplicativos externos, inclusive envolvendo o emprego do formato XML	Alto	Pendente	O Sistema deverá suportar a integração com diferentes sistemas de bioinformática disponíveis na Internet. Como a equipe de desenvolvimento não possui experiência com esse tipo de integração, pode ser que esta necessidade consuma um tempo razoável de desenvolvimento.	Adequar a arquitetura do software para que seja mínimo o impacto na utilização dos diferentes sistemas ou aplicativos. Antecipação do desenvolvimento do mecanismo de comunicação entre o Sistema e os aplicativos externos.
Emprego do formato XML para persistência de dados	Médio	Pendente	Pretende-se que praticamente todas as tabelas de parametrização do Sistema sejam persistidas utilizando XML, para ganhar performance.	Caso a implementação se torne muito complexa, utilizar-se-á o modelo relacional de tabelas.
---	---	---	---	---

Tabela 5.9-1 – Lista de riscos.

5.10 ARQUITETURA CANDIDATA

5.10.1 Organização do sistema em camadas

Para esta fase inicial do desenvolvimento do Sistema, está sendo considerando um modelo bastante simples composto por quatro cadamas, a saber (Figura 5.10-1):

a) Camada de Apresentação: responsável por manter as classes relacionadas com a interface gráfica com o usuário.

b) Camada de Aplicação: é a camada cujas abstrações correspondem às classes de controle (Facade) e que são responsáveis por coordenar a seqüência de atividades necessárias para a realização dos casos de uso.

c) Camada de Domínio do Problema: define o conjunto de classes que são comuns aos diversos aplicativos para o domínio do problema.

d) Camada de Serviços: responsável por encapsular o conjunto de classes responsáveis pela implementação dos mecanismos de persistência e comunicação com os sistemas externos.

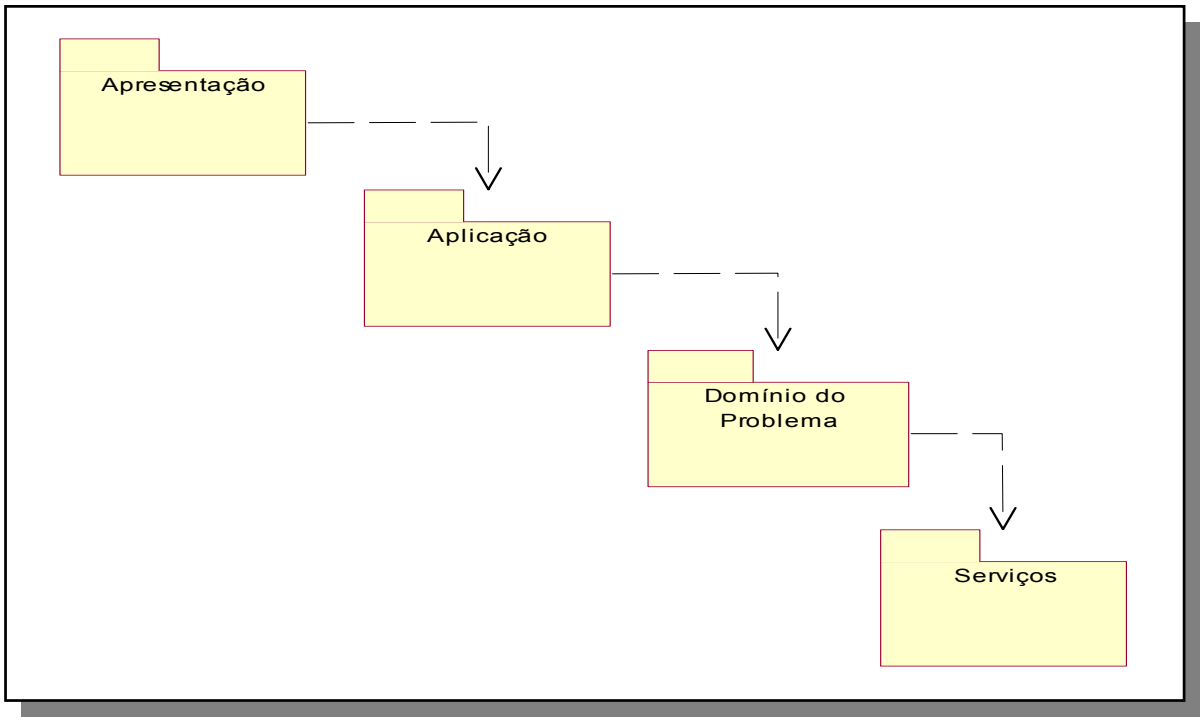

Figura 5.10-1 – Organização do Sistema em camadas.

5.10.2 Mecanismo de persistência

Com o intuito de favorecer a flexibilidade, as classes responsáveis pelo mecanismo de persistência de uma classe de entidade devem estar encapsuladas em subsistemas. Utilizando a linguagem Java, o *design pattern* MVC e o serviço POS de persistência da arquitetura CORBA, resultará o seguinte modelo (Figura 5.10-2):

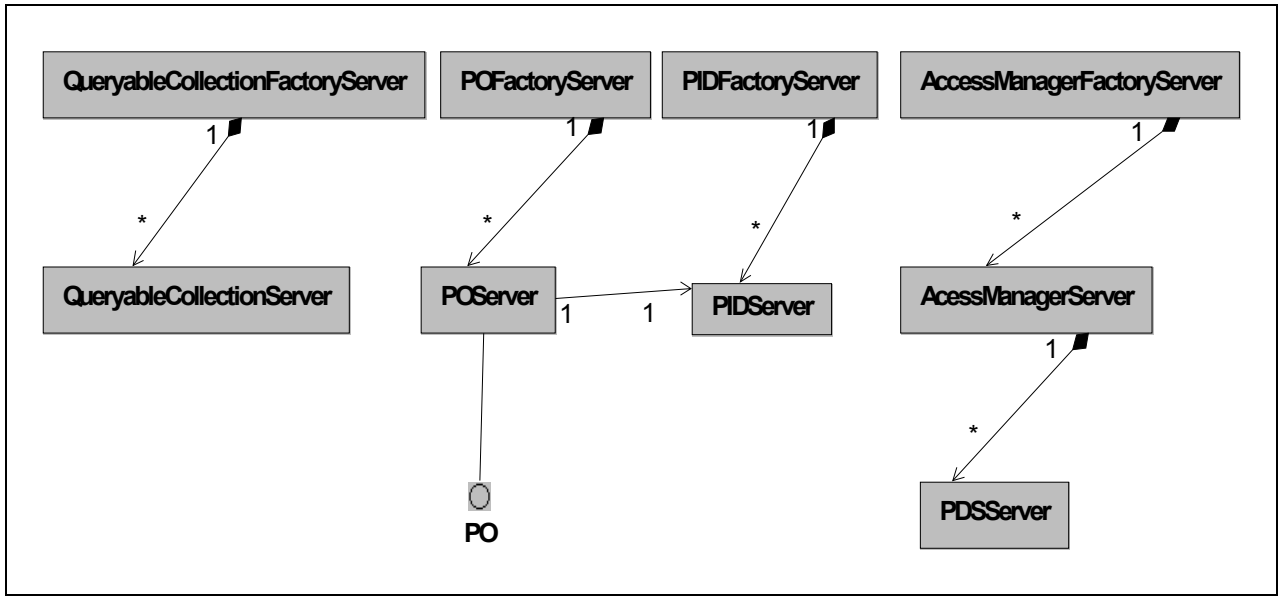

Figura 5.10-2 – Diagrama de classes do mecanismo de persistência do Sistema.

5.10.3 Comunicação com sistemas externos

Da mesma forma que o mecanismo de persistência, as classes responsáveis pela comunicação com sistemas externos devem estar encapsuladas em subsistemas. Como esses sistemas utilizam diferentes protocolos de comunicação, a definição das classes será feita utilizando o *design pattern* Façade para encapsular a complexidade inerente a essa tarefa.

5.11 VISÃO DE CASOS DE USO

5.11.1 Atores

Os principais atores que interagem com o Sistema são:

a) **Pesquisador**: é o especialista em biologia, bioinformática ou qualquer outra área correlata e que, em decorrência de suas atividades em projetos genômicos, utilizará o Sistema como ferramenta auxiliar. Este usuário é um membro da Rede Biofoco e, por isso, possui cadastro regular no Sistema. Além de ser um usuário com amplos privilégios, é também o responsável por todas as atividades, a saber: manutenção dos cadastros, acompanhamento do sistema, emissão de relatórios gerenciais, implementação de melhorias, dentre outras.

b) **Clock**: aplicativo de agendamento de operações e responsável por executar a lista de tarefas do Sistema e realizar consultas baseadas na *wish list*.

c) **Servidor de e-mail**: é o servidor para o qual o Clock enviará o resultado das tarefas executadas, caso o Pesquisador tenha optado receber essa informação via e-mail.

d) ***Sistema Genoma***: o Sistema BLOOM interagirá com este Sistema para proporcionar ao Pesquisador obter o resultado de alinhamento *pairwise,* armazenado no BD *pairwise* XML. Mais especificamente, o Pesquisador acessa o Sistema Genoma, escolhe o item “Consulta”; escolhe o projeto, a biblioteca e a placa; escolhe o item “Análise de seqüência” e, por fim, clica no botão “Resultados do BLAST” para iniciar o acesso ao BLOOM.

e) **BD *pairwise* XML**: seu papel é fornecer os dados armazenados de alinhamento *pairwise* de uma seqüência, obtidos do BLAST inicialmente no formato XML e normalizados para tabelas relacionais. Além disso, é a partir desses dados que serão realizadas as consultas baseadas na *wish list* do pesquisador. Somente o Pesquisador cadastrado pode ter acesso a este

banco de dados. Consulte mais informações no item Interações ou perspectivas do produto.

f) **Módulo de anotação**: seu papel é possibilitar a obtenção dos dados de anotação de cada seqüência apresentada no resultado do alinhamento *pairwise*. Consulte mais informações no item Interações ou perspectivas do produto.

g) **Aplicativo de alinhamento múltiplo**: seu papel é possibilitar a construção do alinhamento múltiplo entre as seqüências escolhidas pelo Pesquisador, quando da visualização gráfica do resultado do alinhamento *pairwise*. Consulte mais informações no item Interações ou perspectivas do produto.

h) **Editor de alinhamento múltiplo**: seu papel é possibilitar a manipulação gráfica do resultado do alinhamento múltiplo. Consulte mais informações no item Interações ou perspectivas do produto.

i) **Aplicativo de predição de estrutura secundária de proteína:** seu papel é possibilitar a predição da estrutura secundária da proteína correspondente à seqüência consenso daquelas escolhidas pelo Pesquisador, quando da visualização gráfica do resultado do alinhamento múltiplo. Consulte mais informações no item Interações ou perspectivas do produto.

j) **Aplicativo de árvore filogenética**: seu papel é possibilitar a derivação da árvore filogenética das seqüências escolhidas pelo Pesquisador, quando da visualização gráfica do resultado do alinhamento múltiplo. Consulte mais informações no item Interações ou perspectivas do produto.

No Diagrama de Classes, no entanto, esses atores não serão apresentados como classes, pois não foram identificados atributos específicos para cada um. **[Booch et al, 2000]** Além disso, embora atores sejam utilizados na modelagem, eles não são, de fato, parte da aplicação, ou seja, apenas interagem com os casos de uso, mas residem fora do sistema.

5.11.2 Casos de uso do ator Pesquisador

a) **Visualizar alinhamento *pairwise***: a partir dos dados de alinhamento *pairwise*, obtidos do BD *pairwise* XML ou diretamenqte do Aplicativo de alinhamento *pairwise*, este caso de uso possibilita ao Pesquisador: obter os dados de anotação das seqüências envolvidas; filtrar o resultado, que pode conter dezenas ou centenas de *hits*, utilizando parâmetros específicos; e, principalmente, visualizar e manipular graficamente o conjunto resultante.

b) **Construir alinhamento múltiplo**: a partir da seleção de um subconjunto das seqüências resultantes do alinhamento *pairwise*, este caso de uso possibilita ao Pesquisador construir e visualizar graficamente o alinhamento múltiplo entre essas.

c) **Predizer estrutura secundária de proteína**: a partir da seleção de um subconjunto das seqüências resultantes do alinhamento múltiplo, este caso de uso possibilita ao Pesquisador executar a ferramenta que prediz e apresenta graficamente a estrutura secundária da proteína consenso, correspondente a cada uma das seqüências de aminoácidos escolhidas.

d) **Derivar árvore filogenética**: a partir da seleção de um subconjunto das seqüências resultantes do alinhamento múltiplo, este caso de uso possibilita ao Pesquisador derivar e visualizar graficamente a árvore filogenética correspondente.

e) **Manter *wish list***: este caso de uso permite ao pesquisador incluir, alterar, excluir e consultar os assuntos de seu interesse, a partir do registros mantidos nessa lista.

f) **Manter tarefa**: este caso de uso permite ao pesquisador incluir, alterar, excluir e consultar as tarefas a serem executadas automaticamente pelo Sistema.

Na Figura 5.11-1 a seguir, é apresentado o diagrama dos casos de usos do ator Pesquisador :

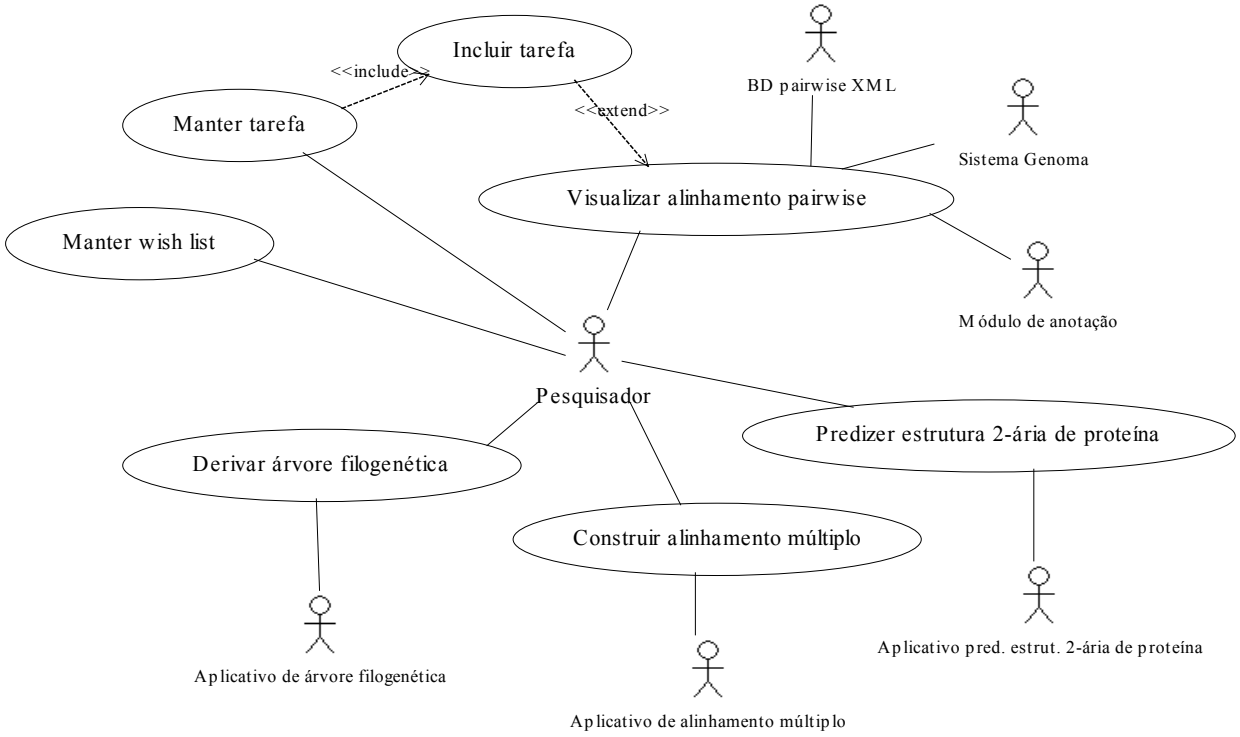

Figura 5.11-1 – Diagrama dos Casos de Uso do ator Pesquisador

5.11.3 Casos de uso do ator Clock

a) Executar tarefa: a partir dos dados da tarefa agendada, o Sistema a executa automaticamente na data prevista, salva o resultado no BD *pairwise* XML e pode enviar, por email, um aviso ao pesquisador de que a tarefa foi cumprida.

b) Realizar consulta baseada na *wish list*: a partir dos dados constantes na *wish list* de cada pesquisador, o Sistema periodica e automaticamente consulta o BD *pairwise* XML em busca de informações mais atualizadas que satisfaçam as preferências registradas. Em seguida, pode enviar ao pesquisador, por email, o resultado da consulta.

Na Figura 5.11-2 a seguir, é apresentado o diagrama dos casos de usos do ator Clock:

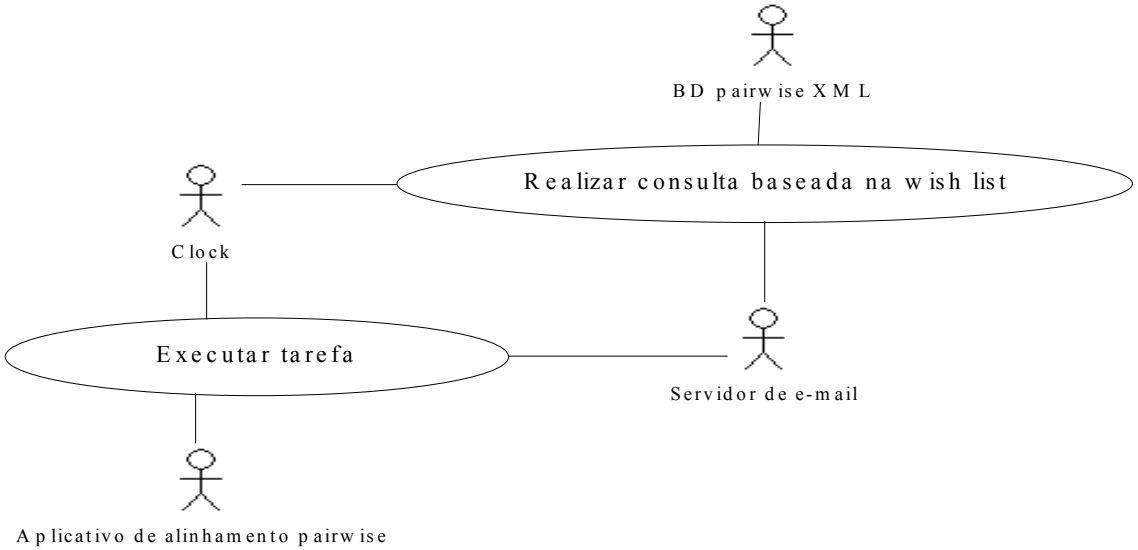

Figura 5.11-2 – Diagrama dos Casos de Uso do ator Clock

5.11.4 Prioridade dos casos de uso

Os casos de uso já levantados serão priorizados de acordo com os riscos, a necessidade dos usuários e a complexidade de desenvolvimento. A priorização dos casos de uso serve de entrada para a elaboração do plano de desenvolvimento de software. Serão utilizados os valores "1-Alto", "2-Médio" e "3-Baixo" para qualificar os atributos riscos, necessidade e complexidade, conforme Tabela 5.11-1. A cada iteração é necessário rever a prioridade dos casos de uso.

Caso de Uso	Necessidade	Complexidade	Risco
Visualizar alinhamento *pairwise*	1-Alta	1-Alta	1-Alto
Construir alinhamento múltiplo	1-Alta	1-Alta	2-Médio
Derivar árvore filogenética	2-Média	2-Média	2-Médio
Predizer estrutura secundária de proteína	2-Média	2-Média	2-Médio
Executar tarefa	3-Média	2-Média	3-Baixo
Manter *wish list*	3-Baixa	2-Média	3-Baixo
Realizar consulta baseada na *wish list*	3-Baixa	2-Média	3-Baixo
Incluir tarefa	3-Baixa	3-Baixa	3-Baixo
Manter tarefa	3-Baixa	3-Baixa	3-Baixo

Tabela 5.11-1 – Prioridade dos casos de uso.

5.11.5 Detalhamento de alguns casos de uso

I. Visualizar alinhamento *pairwise*

a) *Breve descrição*: a partir dos dados de alinhamento *pairwise*, obtidos do BD *pairwise* XML por intermédio do Sistema Genoma, este caso de uso possibilita ao Pesquisador: obter os dados de anotação das seqüências envolvidas; filtrar o resultado, que pode conter dezenas ou centenas de *hits*, utilizando parâmetros específicos; e, principalmente, visualizar e manipular graficamente todo o conjunto resultante.

b) *Pré-condições*:

1. Acessar localmente o Sistema. Caso a versão esteja desatualizada, o aplicativo Java Web Start se encarrega de baixar a nova versão automaticamente. Se for o primeiro acesso, o pesquisador deve recorrer à página do Laboratório de Bioinformática, baixar a aplicação para sua máquina local e, então, iniciar o uso.

c) *Fluxo principal de eventos*:

1. Clicar duas vezes no ícone do Sistema Genoma.
2. Informar login e senha.
3. Escolher a opção “Consulta”.
4. O pesquisador seleciona, a partir dos registros disponíveis no BD *pairwise* XML, o projeto, a biblioteca e a placa.
5. Escolher a opção “Análise de seqüência”.
6. Escolher um dos clones apresentados na lista.
7. Feito isso, o Sistema recupera o resultado armazenado correspondente a esse clone (ou seqüência).
8. Automaticamente, o Sistema interage com o Módulo de anotação e faz com que esse obtenha os dados de anotação de cada seqüência envolvida.
9. Reunidas essas informações, o Sistema constrói a apresentação gráfica e permite que o usuário interaja com o novo resultado, por exemplo: filtrando registros por parâmetros específicos, ampliando ou reduzindo o “*zoom*” dos objetos, selecionado subconjunto de seqüências, alterando a visualização para o hit selecionado etc.

10. Se desejar executar outro aplicativo em seguida, a partir da tela atual, um subconjunto de seqüências deve ser selecionado clicando diretamente no hit e, em seguida, no botão “Adicionar”.
11. Para executar o próximo aplicativo desejado, seleciona um item dentre aqueles constantes na lista de funcionalidades, por exemplo alinhamento múltiplo, predição de estrutura secundária de proteína etc.
12. Se não desejar prosseguir, basta clicar no botão “Retornar” para retornar à tela anterior, que é a de Análise de seqüência.

d) ***Fluxo alternativo de eventos***:

2a. O sistema detecta que o login não está cadastrado, emite uma mensagem de alerta ao usuário e o instrui a procurar o administrador da rede ou do sistema.

4a. O sistema não prossegue e emite um alerta de que não há resultado XML armazenado para a seqüência escolhida. O usuário pode tentar outras seqüências.

8a. O sistema não prossegue e emite um alerta de que o usuário se esqueceu de selecionar o subconjunto de seqüências.

e) ***Ponto de extensão***:

O pesquisador pode também solicitar, para uma data posterior, a reexecução desse alinhamento pelo agendamento da tarefa (`extend Incluir tarefa`).

f) ***Pós-condições***:

1. Se o pesquisador optou por agendar uma tarefa, essa transação deve ser finalizada com sucesso.
2. Um subconjunto de seqüências deve estar selecionado para que seja possível executar outros aplicativos em seguida, a partir da tela atual.

Segue o diagrama de atividades correspondente (Figura 5.11-3):

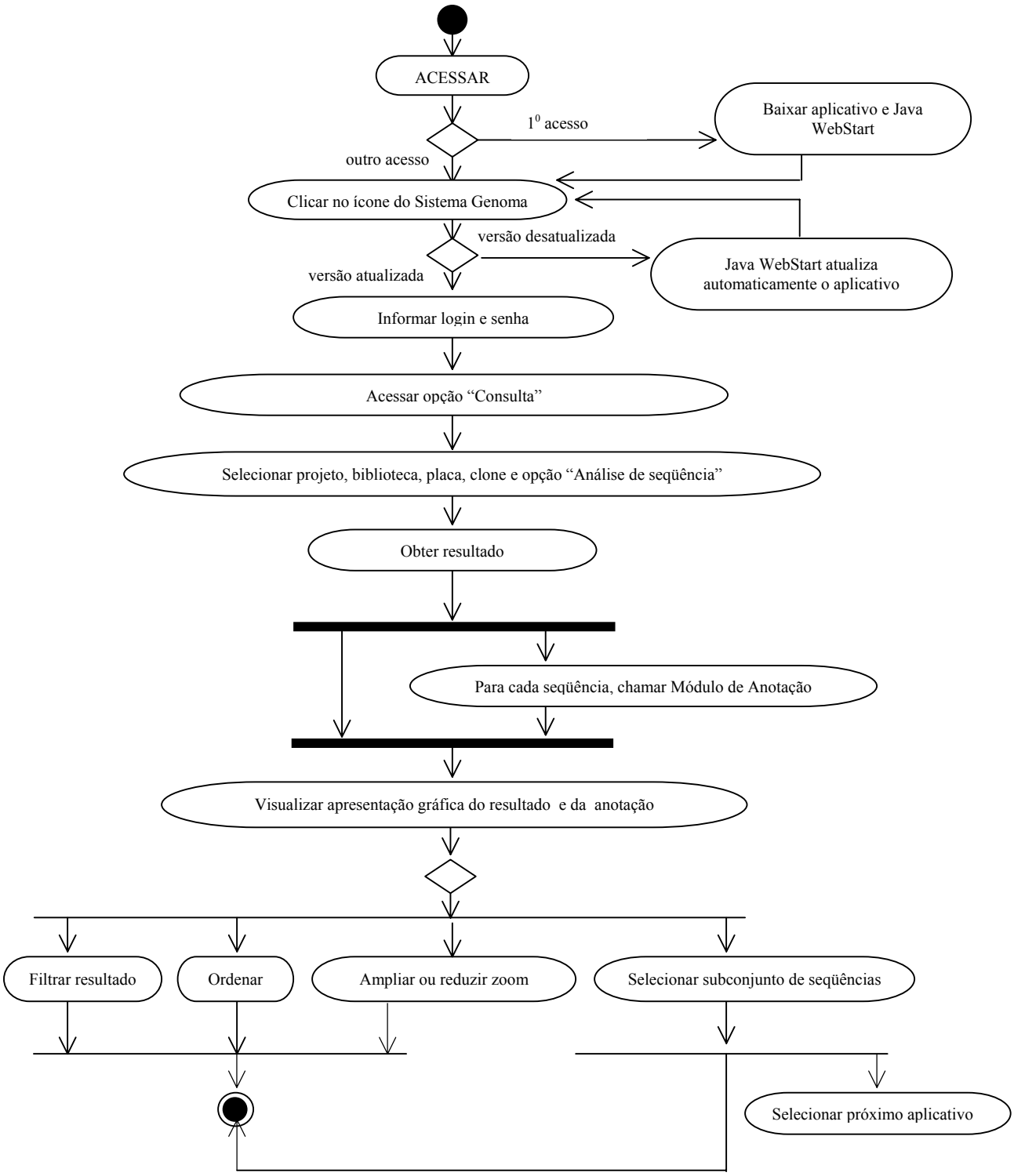

Figura 5.11-3 – Diagrama de Atividades – Visualizar alinhamento *pairwise.*

II. Construir alinhamento múltiplo

a) ***Breve descrição***: a partir da seleção de um subconjunto das seqüências resultantes do alinhamento *pairwise*, este caso de uso possibilita ao Pesquisador executar a ferramenta que constrói o alinhamento múltiplo correspondente ao subconjunto selecionado. Em seguida, pode-se visualizar e manipular graficamente todo o conjunto resultante.

b) ***Pré-condições***:

1. Ter selecionado um subconjunto das seqüências resultantes do alinhamento *pairwise.*

c) ***Fluxo principal de eventos***:

1. Este caso de uso se inicia quando o pesquisador seleciona um aplicativo para executar a atividade, dentre aqueles constantes na lista de ferramentas apropriadas.
2. Clica no botão “Executar”.
3. O Sistema utiliza os parâmetros pré-determinados (*default)* de cada ferramenta.
4. Feito isso, o Pesquisador atribui uma descrição breve e um comentário para o novo registro.
5. Determina o destino do resultado: mostrar na tela; gravar no caminho e arquivo especificados; gravar diretamente no BD *pairwise* XML; ou todas as opções.
6. Clica no botão “OK” e o Sistema executa a ferramenta. O usuário deve aguardar até o término da transação.
7. Se o pesquisador escolheu mostrar resultado na tela, o Sistema automaticamente chamará o Editor de alinhamento múltiplo, o qual permitirá, então, que o usuário interaja graficamente com os dados obtidos.

13. Se desejar executar outro aplicativo em seguida, a partir da tela atual, um subconjunto de seqüências deve ser selecionado clicando diretamente no hit e, em seguida, no botão “Adicionar”.
14. Para executar o próximo aplicativo desejado, seleciona um item dentre aqueles constantes na lista de funcionalidades, por exemplo derivação de árvore filogenética etc.

15. Se não desejar prosseguir, basta clicar no botão “Retornar” para retornar à tela anterior, que é a do alinhamento *pairwise*.

d) *Fluxo alternativo de eventos*:

1a. O pesquisador pode, opcionalmente, expandir o subconjunto de seqüências já selecionadas, agregando outras externas. Basta clicar no botão “Agregar seqüências”, informar cabeçalho e escrever ou colar a seqüência na segunda área de texto.

3a. O pesquisador clica no botão “Parâmetros” e pode alterar os parâmetros específicos de execução da ferramenta.

8a. O sistema não prossegue e emite um alerta de que o usuário se esqueceu de selecionar o subconjunto de seqüências.

e) *Pós-condições*:

1. Se o pesquisador optou por salvar o resultado em arquivo ou no banco de dados, essa transação deve ser finalizada com sucesso.
2. Um subconjunto de seqüências deve estar selecionado para que seja possível executar outros aplicativos em seguida, a partir da tela atual.

Segue o diagrama de atividades correspondente (Figura 5.11-4):

[Vide próxima página]

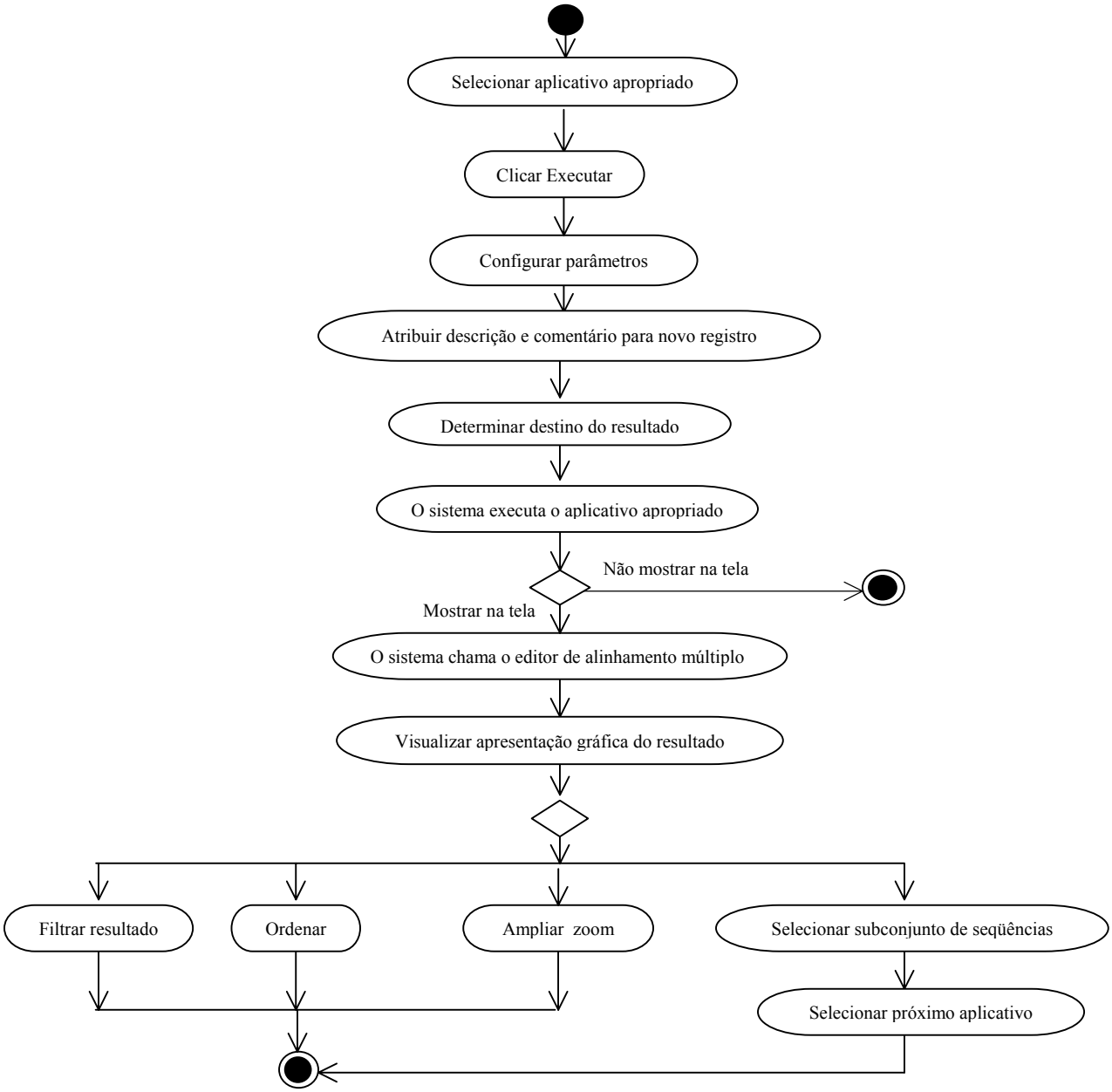

Figura 5.11-4 – Diagrama de Atividades – Construir alinhamento múltiplo.

III. Predizer estrutura secundária de proteína

a) ***Breve descrição***: a partir da seleção de um subconjunto das seqüências resultantes do alinhamento múltiplo, este caso de uso possibilita ao Pesquisador executar a ferramenta que prediz a estrutura secundária da proteína correspondente à seqüência consenso do subconjunto selecionado. Em seguida, pode-se visualizar e manipular graficamente todo o conjunto resultante.

b) *Pré-condições*:

1. Ter selecionado um subconjunto das seqüências resultantes do alinhamento múltiplo.
2. Todas as seqüências selecionadas devem ser de aminoácidos (proteína).

c) *Fluxo principal de eventos*:

1. Este caso de uso se inicia quando o pesquisador seleciona um aplicativo para executar a atividade, dentre aqueles constantes na lista de ferramentas apropriadas.
2. Clica no botão “Executar”.
3. O Sistema utiliza os parâmetros pré-determinados (*default)* de cada ferramenta.
4. Feito isso, o Pesquisador atribui uma descrição breve e um comentário para o novo registro.
5. Determina o destino do resultado: mostrar na tela; gravar no caminho e arquivo especificados; gravar diretamente no BD *pairwise* XML; ou todas as opções.
6. Clica no botão “OK” e o Sistema executa a ferramenta. O usuário deve aguardar até o término da transação.
7. Se o pesquisador escolheu mostrar resultado na tela, o Sistema permitirá, então, que o usuário interaja graficamente com os dados obtidos.
8. Ao concluir, clica no botão “Retornar” para retornar à tela anterior, que é a do alinhamento múltiplo.

d) *Fluxo alternativo de eventos*:

3a. O pesquisador clica no botão “Parâmetros” e pode alterar os parâmetros específicos de execução da ferramenta.

e) *Pós-condições*:

1. Se o pesquisador optou por salvar o resultado em arquivo ou no banco de dados, essa transação deve ser finalizada com sucesso.

Segue o diagrama de atividades correspondente (Figura 5.11-5):

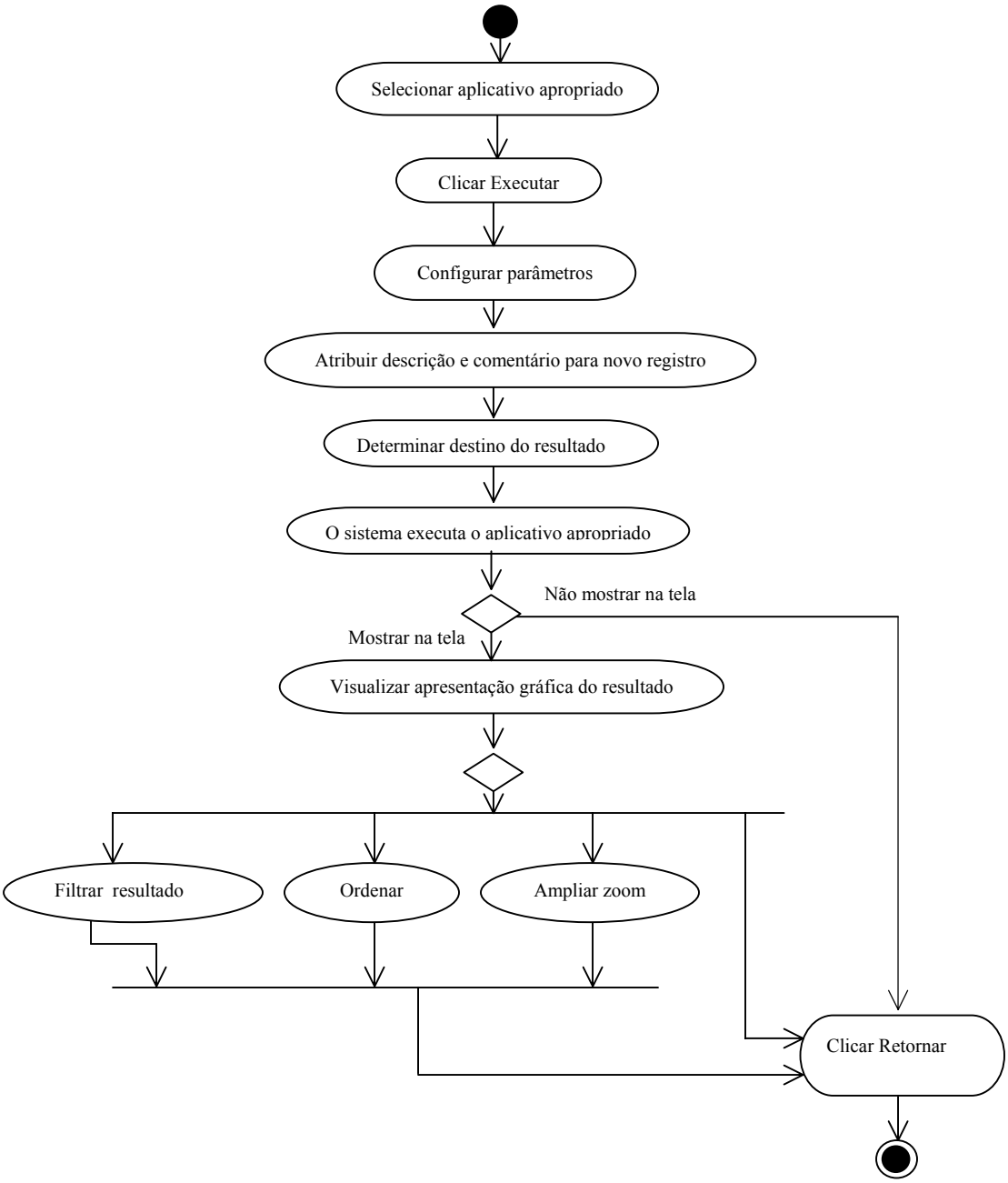

Figura 5.11-5 – Diagrama de Atividades – Predizer estrutura secundária de proteína.

IV. Derivar árvore filogenética

a) ***Breve descrição***: a partir da seleção de um subconjunto das seqüências resultantes do alinhamento múltiplo, este caso de uso possibilita ao Pesquisador executar a ferramenta que deriva a árvore filogenética correspondente ao subconjunto selecionado. Em seguida, pode-se visualizar e manipular graficamente todo o conjunto resultante.

b) ***Pré-condições***:

1. Ter selecionado um subconjunto das seqüências resultantes do alinhamento múltiplo.

c) ***Fluxo principal de eventos***:

1. Este caso de uso se inicia quando o pesquisador seleciona um aplicativo para executar a atividade, dentre aqueles constantes na lista de ferramentas apropriadas.
2. Clica no botão “Executar”.
3. O Sistema utiliza os parâmetros pré-determinados (*default)* de cada ferramenta.
4. Feito isso, o Pesquisador atribui uma descrição breve e um comentário para o novo registro.
5. Determina o destino do resultado: mostrar na tela; gravar no caminho e arquivo especificados; gravar diretamente no BD *pairwise* XML; ou todas as opções.
6. Clica no botão “OK” e o Sistema executa a ferramenta. O usuário deve aguardar até o término da transação.
7. Se o pesquisador escolheu mostrar resultado na tela, o Sistema permitirá, então, que o usuário interaja graficamente com os dados obtidos. A árvore resultante é apresentada conforme o tipo default de visualização ou aquele escolhido previamente.
8. Ao concluir, clica no botão “Retornar” para retornar à tela anterior, que é a do alinhamento múltiplo.

d) ***Fluxo alternativo de eventos***:

3a. O pesquisador clica no botão “Parâmetros” e pode alterar os parâmetros específicos de execução da ferramenta.

7a. Essa visualização não é estática e permite-se ao pesquisador intercambiar entre os diversos tipos disponíveis, clicando no botão “Outro tipo de árvore”.

7b. Se julgar necessário, o pesquisador clica no botão “Exportar” para exportar a árvore resultante como um arquivo gráfico (formatos texto e PDF), para que seja utilizado em publicações.

e) *Pós-condições*:

1. Se o pesquisador optou por salvar o resultado em arquivo ou no banco de dados, essa transação deve ser finalizada com sucesso.

Segue o diagrama de atividades correspondente (Figura 5.11-6):

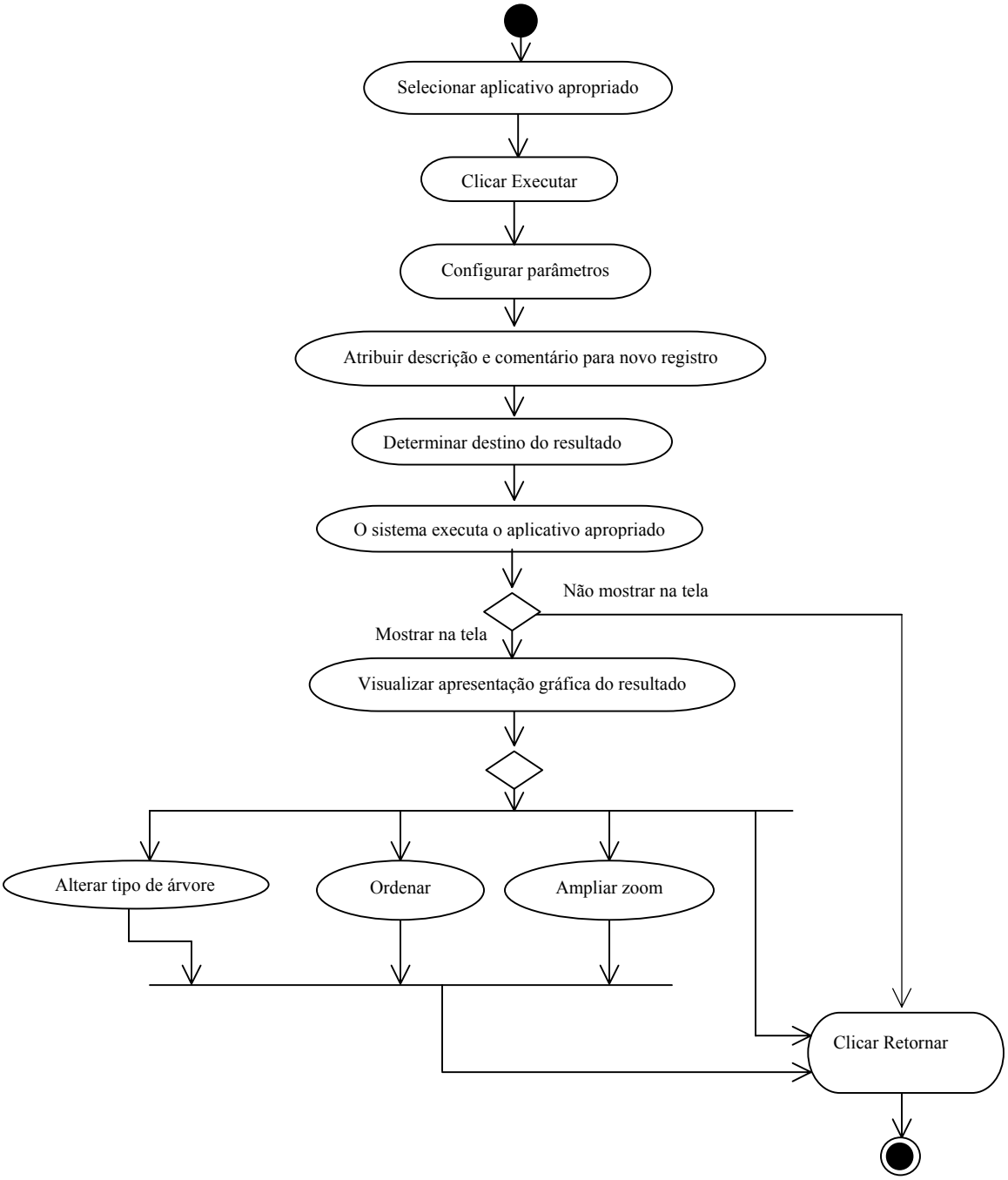

Figura 5.11-6 – Diagrama de Atividades – Derivar árvore filogenética.

5.12 DISTRIBUIÇÃO DAS ITERAÇÕES

Devido à utilização do Processo Unificado Rational, as quatro fases e suas respectivas iterações estão distribuídas conforme a Tabela 5.12-1 a seguir:

Fase	Iteração	Início	Fim	Atividade
Concepção	I1	01/09/02	30/09/02	- Identificação do problema do usuário. - Identificação das necessidades do usuário. - Análise da oportunidade científica do produto. - Elaboração do plano de desenvolvimento de software. - Análise dos resultados da iteração.
	I2	01/10/02	31/10/02	- Identificação de uma arquitetura candidata. - Descrição da comunicação com diferentes sistemas. - Detalhamento do caso de uso Visualizar alinhamento *pairwise.* - Análise dos resultados da iteração. - Planejar a fase de Elaboração.
Elaboração	E1	01/11/02	18/12/02	- Detalhamento do caso de uso Construir alinhamento múltiplo. - Detalhamento do caso de uso Predizer estrutura secundária de proteína. - Detalhamento do caso de uso Derivar árvore filogenética. - Análise e projeto dos serviços de filtragem e de parametrização. - Análise dos resultados da iteração.
	E2	01/01/03	20/02/03	- Análise, projeto, implementação e testes do caso de uso Visualizar alinhamento *pairwise.* - Análise dos resultados da iteração. - Planejamento da fase de Construção.
Construção	C1	21/02/03	30/04/03	- Análise, projeto, implementação e testes do caso de uso Construir alinhamento múltiplo. - Análise dos resultados da iteração.
	C2	01/05/03	15/06/03	- Implementação e testes dos serviços de filtragem e de parametrização. - Análise dos resultados da iteração.
	C3	16/06/03	31/07/03	- Implementação e testes das classes persistentes em XML dos serviços de filtragem e de parametrização, ou seja, *os atributos desses serviços serão mantidos em arquivos XML e gravados no .jar de download da aplicação.* - Análise dos resultados da iteração.
	C4	01/08/03	01/10/03	- Análise, projeto, implementação e testes do caso de uso Predizer estrutura secundária de proteína. - Análise dos resultados da iteração.
	C5	02/10/03	02/12/03	- Análise, projeto, implementação e testes do caso de uso Derivar árvore filogenética. - Análise dos resultados da iteração.

	C6	03/12/03	03/02/04	- Detalhamento do caso de uso Incluir tarefa. - Análise, projeto, implementação e testes do caso de uso Incluir tarefa. - Detalhamento do caso de uso Manter tarefa. - Análise, projeto, implementação e testes do caso de uso Manter tarefa. - Detalhamento do caso de uso Executar tarefa. - Análise, projeto, implementação e testes do caso de uso Executar tarefa. - Análise dos resultados da iteração.
	C7	04/02/04	04/04/04	- Detalhamento do caso de uso Manter *wish list.* - Análise, projeto, implementação e testes do caso de uso Manter *wish list.* - Detalhamento do caso de uso Realizar consulta baseada na *wish list.* - Análise, projeto, implementação e testes do caso de uso Realizar consulta baseada na *wish list.* - Análise dos resultados da iteração. - Planejamento da fase de Transição.
Transição	T1	05/04/04	30/04/04	- Configuração do ambiente de produção. - Instalação e testes do Sistema em ambiente de produção. - Homologação do Sistema.

Tabela 5.12-1 – Distribuição das iterações.

5.13 VISÃO DE PROJETO

5.13.1 Diagramas de classes

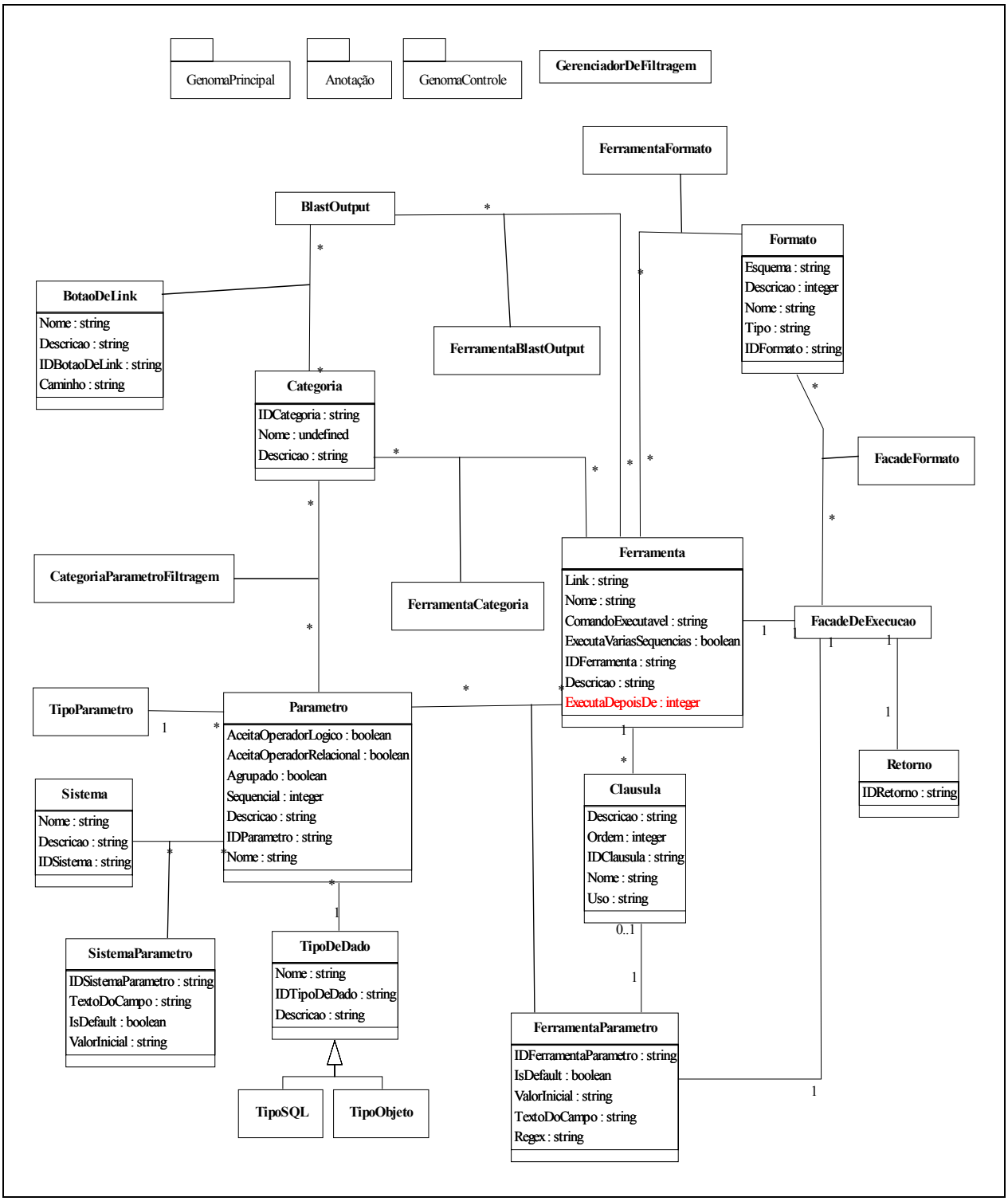

Figura 5.13-1 – Sistema BLOOM: diagrama de classes dos serviços parametrização e filtragem.

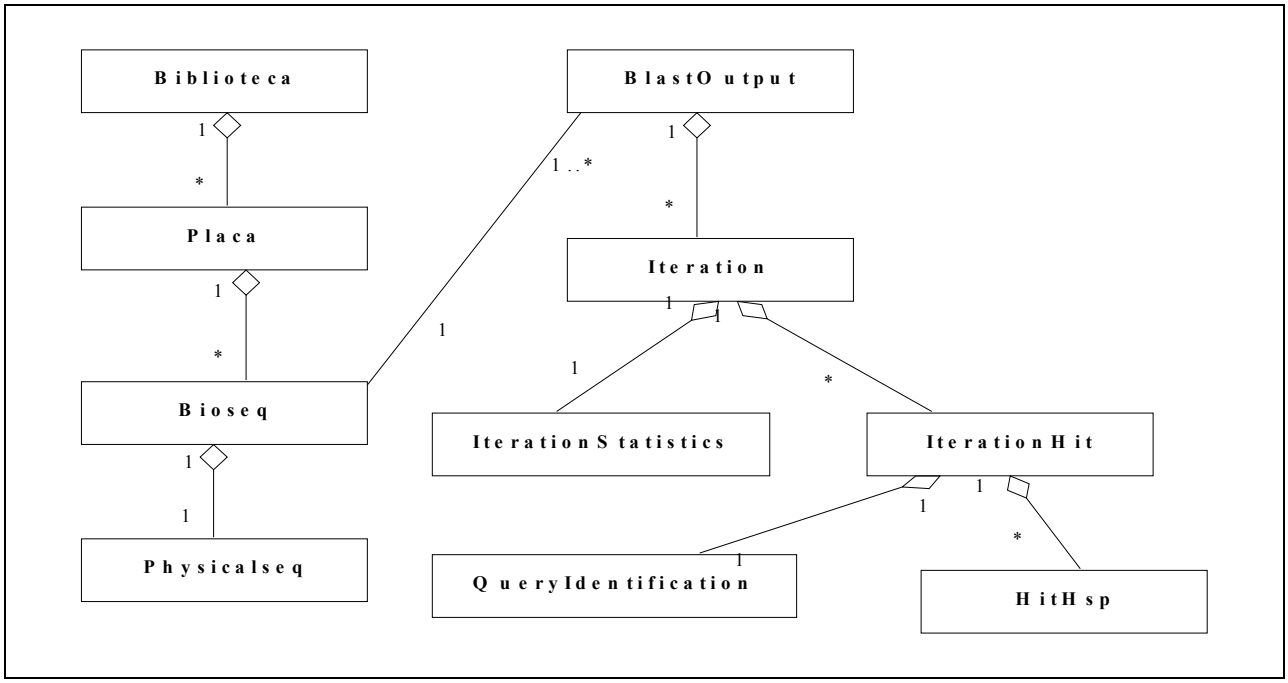

Figura 5.13-2 – Sistemas Genoma e BLOOM: diagrama de classes de resultado do BLAST (comuns a ambos). Originalmente esses conjuntos de dados chegam no formato XML e passam pelo processo de *unmarshaling* no Sistema Genoma para se tornarem objetos.

5.13.2 Diagramas de interação

I. Diagramas de seqüência

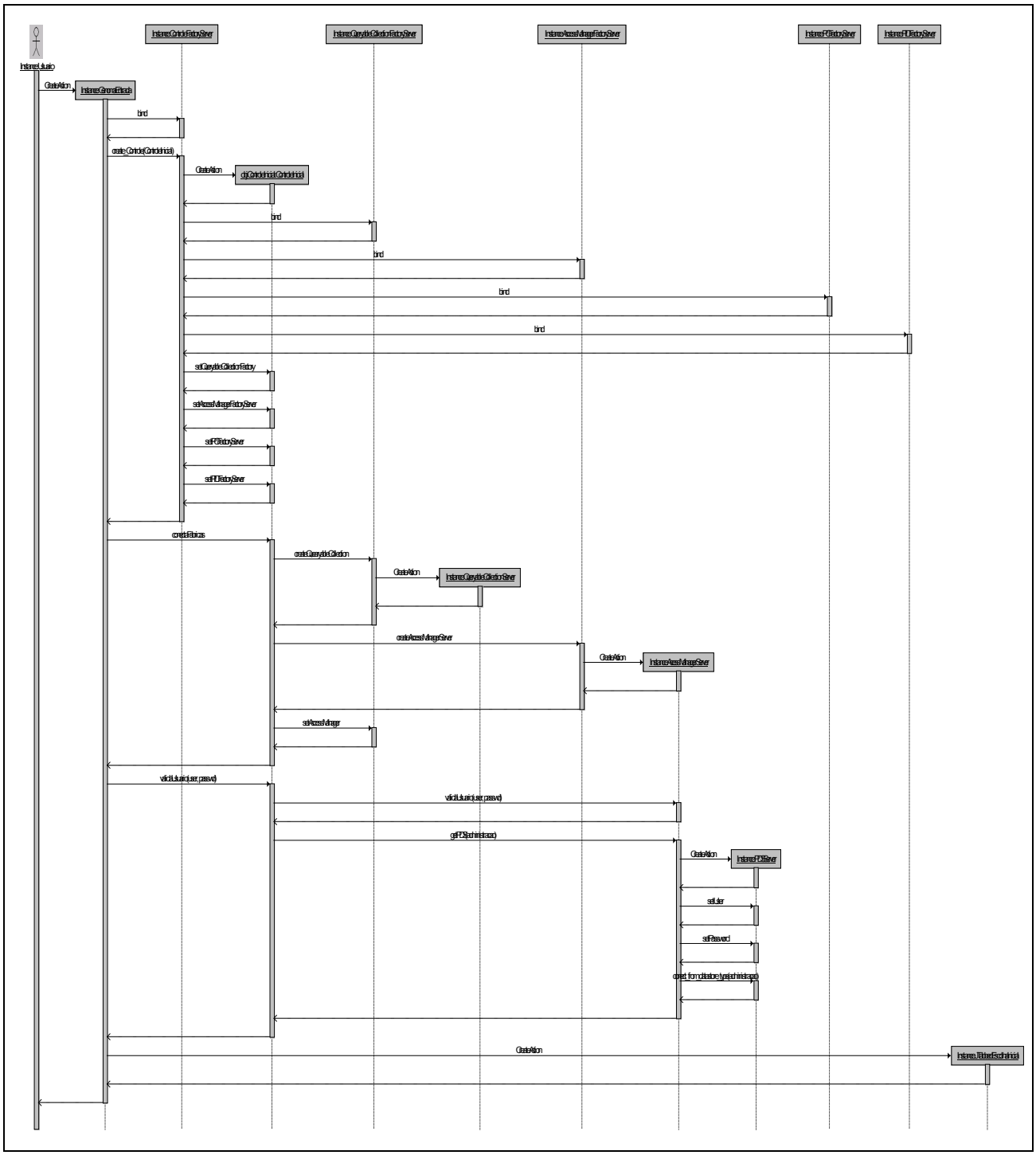

Figura 0-1 – Sistema Genoma: diagrama de seqüência do cenário Inicializa.

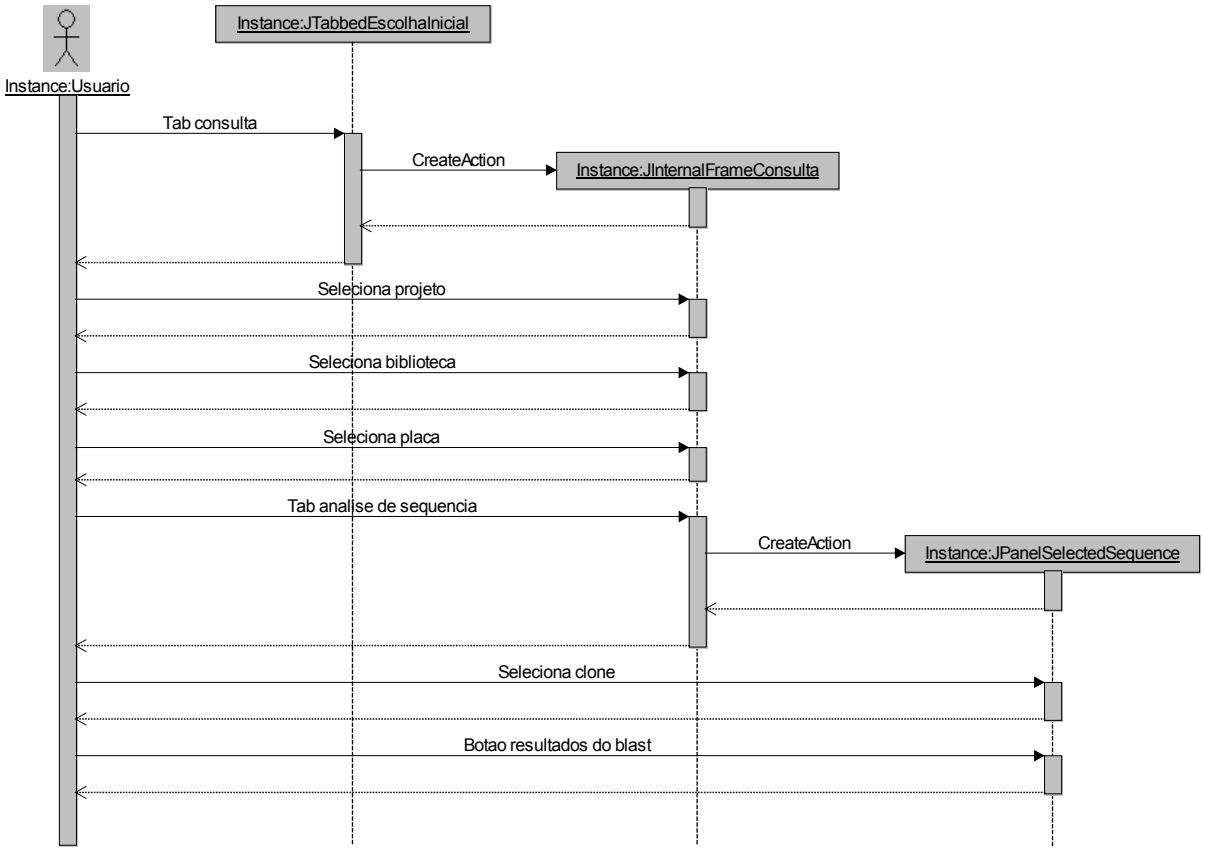

Figura 0-2 – Sistema Genoma: diagrama de seqüência do cenário Resultados do BLAST.

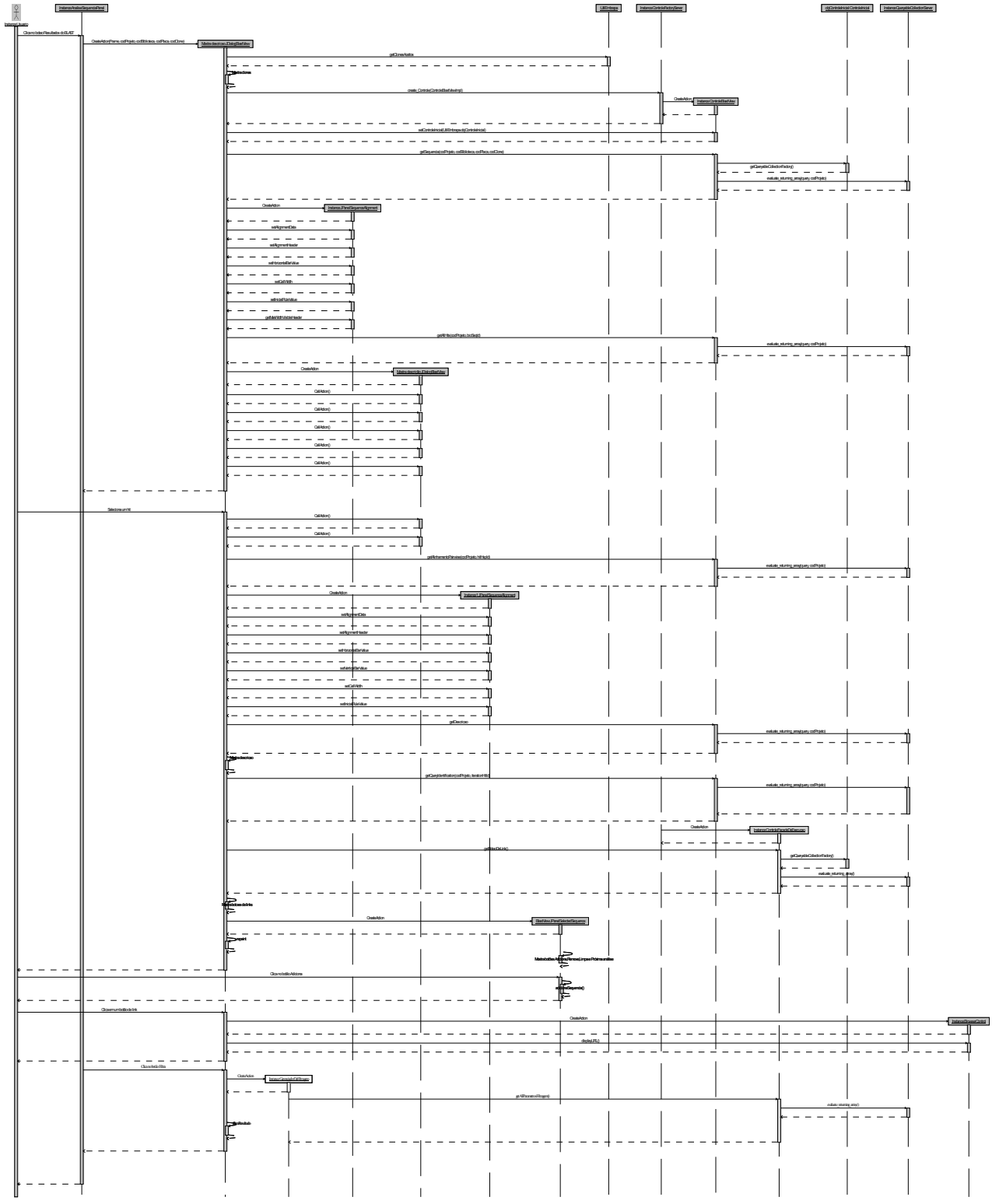

Figura 0-1 – Sistema BLOOM: diagrama de seqüência do cenário Visualizar alinhamento *pairwise*.

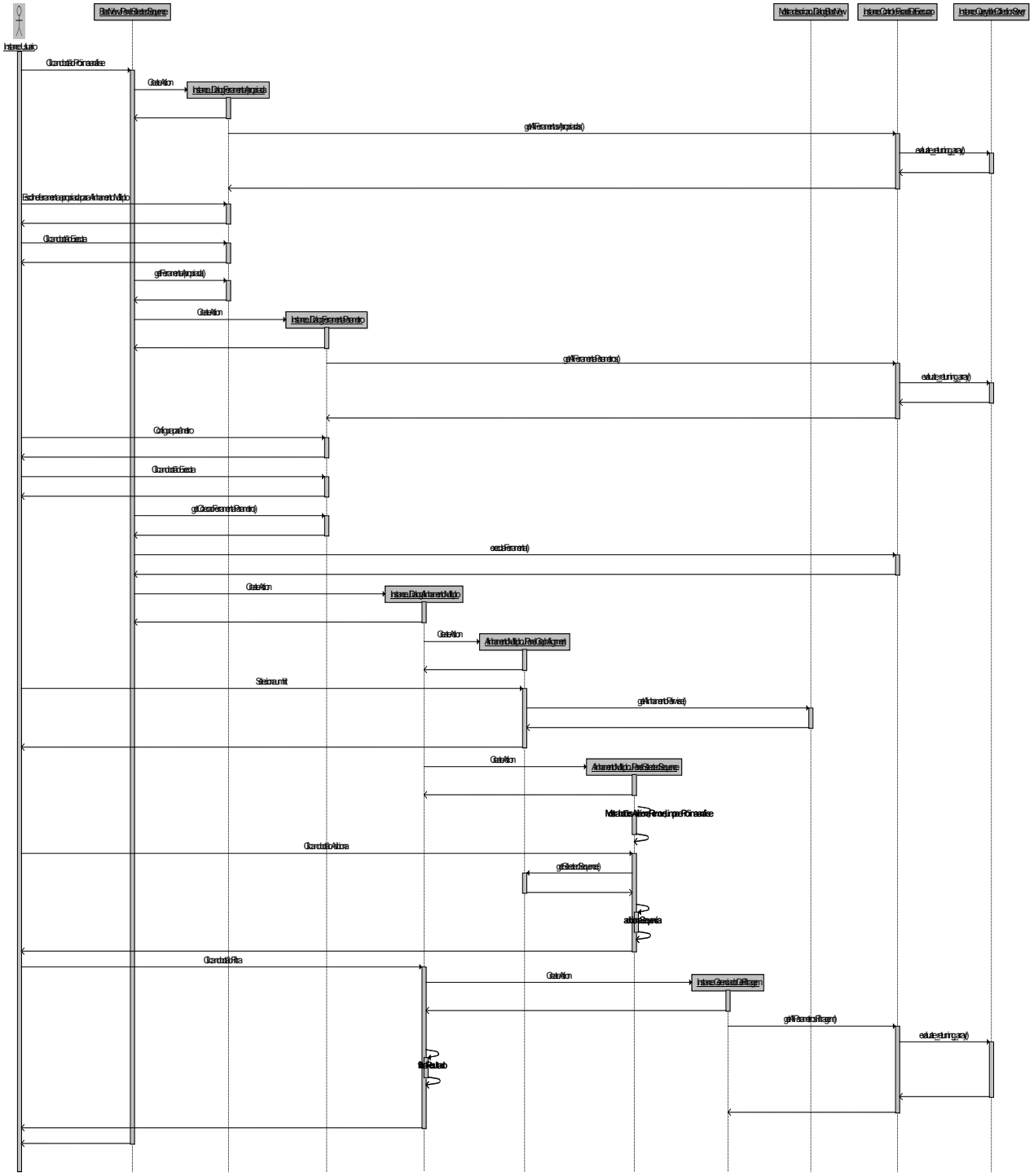

Figura 0-1 – Sistema BLOOM: diagrama de seqüência do cenário Construir alinhamento múltiplo.

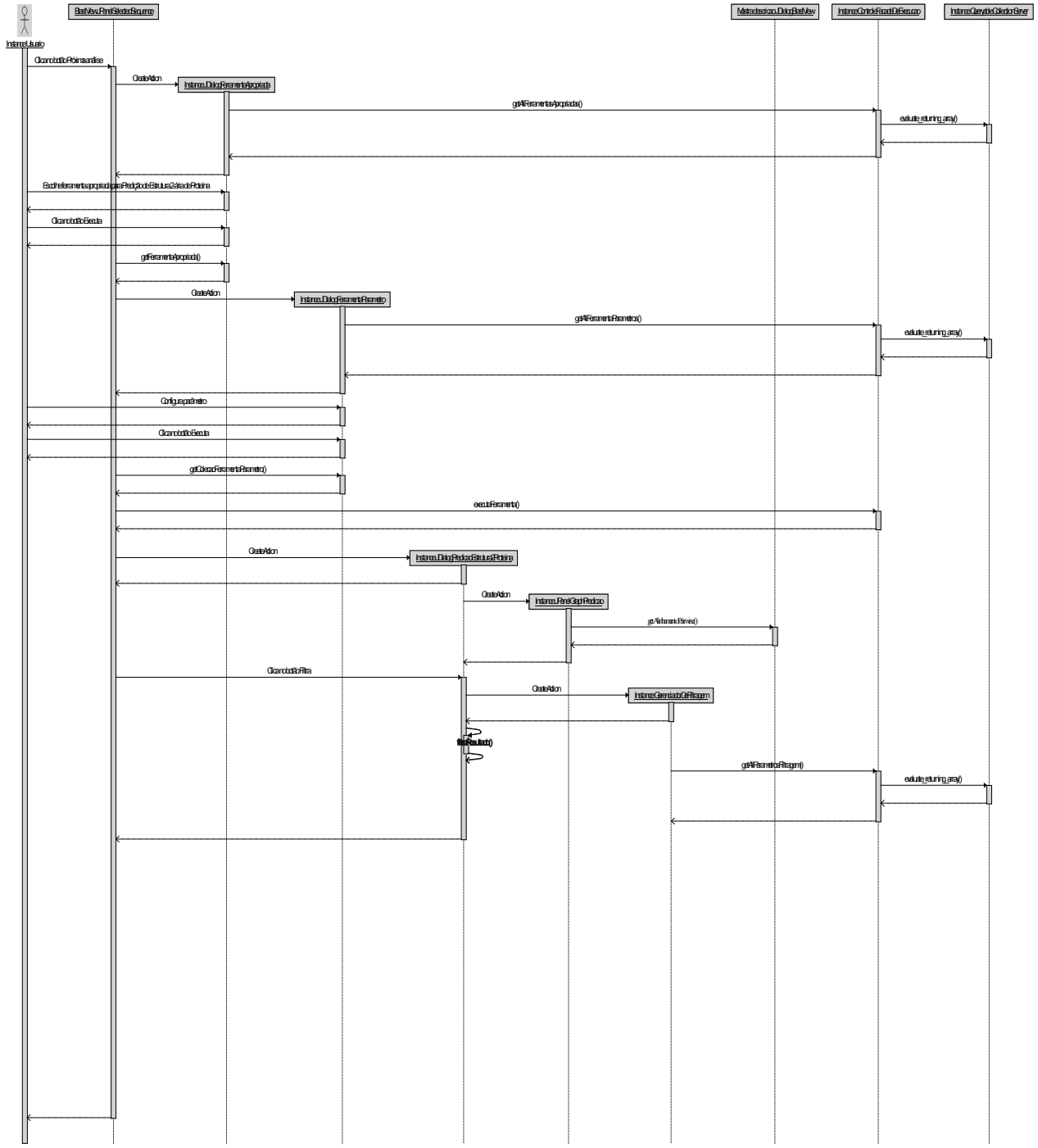

Figura 0-1 – Sistema BLOOM: diagrama de seqüência do cenário Predizer estrutura secundária de proteína.

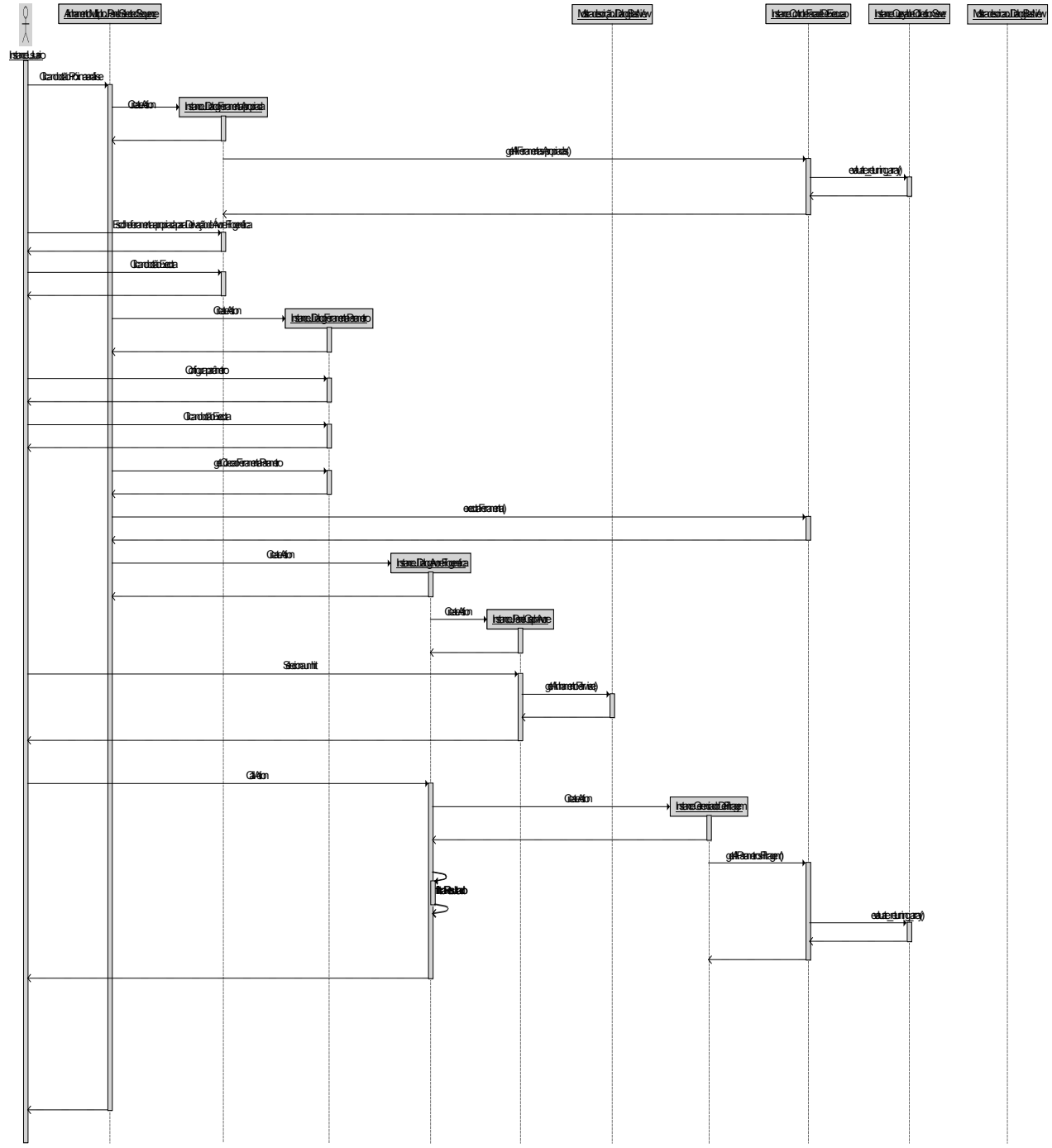

Figura 0-2 – Sistema BLOOM: diagrama de seqüência do cenário Derivar árvore filogenética.

6. RESULTADOS

O Sistema está parcialmente implementado, contemplando o cenário principal do use case Visualizar alinhamento *pairwise* e o início da integração com Visualizar alinhamento múltiplo. Com isso, já se pôde testar a integração com o BD *pairwise* XML e com o Sistema Genoma. O restante da implementação seguirá o plano de desenvolvimento.

Embora a implementação esteja ainda na primeira iteração da fase de Construção (*Construction* do RUP), já é possível assinalar os resultados ou ganhos tanto para a Bioinformática quanto para a Informática.

Para a Bioinformática, os resultados são:

a) Como primeira parte da pesquisa, foi necessário escrever um Tutorial de biologia molecular computacional, o qual tornou-se o capítulo 3 desta Dissertação. Esse Tutorial será divulgado como conhecimento no portal da Biofoco.

b) Atendendo às necessidades dos usuários, o aplicativo desenvolvido é gráfico, interativo, de código aberto – respeitadas as propriedades intelectuais - e de livre acesso a todos os clientes e usuários da rede Biofoco.

c) O pesquisador já pode fazer a experimentação da mesma análise em aplicativos diferentes, escolhidos a partir de uma lista.

d) O pesquisador também já pode executar uma análise em seguida a outra, fazendo com que o resultado da anterior trafegue transparentemente como entrada para a próxima análise, sem qualquer esforço adicional, por exemplo: alinhamento múltiplo e, em seguida, derivação de árvore filogenética do conjunto de seqüências selecionadas no alinhamento.

e) Devido à necessidade de integrar, pelo menos, os três sistemas previstos no Projeto Biofoco, criou-se um ambiente padronizado de acesso, o qual está propiciando melhor gerenciamento, organização e disponibilização das informações. Essa estratégia é a mesma que vinha sendo seguida pelo Sistema Genoma.

f) Para evitar perda de tempo e esforço do pesquisador com execução manual de atividades, foram projetados a *wish list* e o agendamento das tarefas que podem ser executadas automaticamente pelo Sistema.

Para a Informática, os resultados são:

a) Já estão concluídos, até a fase atual, o levantamento de necessidades do usuário e o projeto de integração com vários aplicativos. Isso facilita a integração de novos desenvolvedores à equipe.

b) Foram produtivos a validação e o emprego de orientação a objetos, *design patterns*, JAVA, CORBA, UML e RUP no desenvolvimento de uma ferramenta para uso científico.

c) O desenvolvimento do aplicativo foi facilitado pelo reuso integral da arquitetura empregada no Sistema Genoma, principalmente quanto aos serviços de Controle, Coleção e Persistência. Com isso, mesmo os desenvolvedores que não detinham conhecimento de todas as tecnologias empregadas, principalmente CORBA, construíram, de imediato, componentes úteis.

d) A implementação e posteriores manutenções tornaram-se bastante facilitadas pelo emprego de atributos parametrizados.

e) Os Serviços de Parametrização e de Filtragem de resultados são componentes de uso genérico e suas facilidades serão estendidas aos Sistemas Genoma e Anotação.

7. CONCLUSÃO

Quanto à pesquisa realizada e ao Sistema em fase inicial de implementação, pode-se concluir que :

a) O informata precisa ter um conhecimento mínimo de biologia molecular computacional para conseguir propor soluções e entender a demanda dos pesquisadores para este Projeto. E esse conhecimento ocorreu na pesquisa empreendida para escrever o Referencial Teórico (capítulo 3 desta Dissertação).

b) O uso do Sistema como ferramenta de *datamining* auxilia na visualização e exploração das informações que trafegam como entrada ou saída entre as diversas ferramentas integradas.

c) O ambiente proposto para integração das ferramentas mostra-se bastante complexo. Por isso, não se pode prescindir do uso de um processo de desenvolvimento de software com boa documentação em todas as fases; emprego de tecnologias atuais de objetos distribuídos; e constante adoção de soluções padronizadas de projeto e integração, representadas por *design patterns* e XML, respectivamente.

d) Ainda não há no mercado e nem no meio acadêmico um ambiente com o nível de integração proposto neste projeto, conforme pesquisas realizadas até o mês de novembro de 2002.

e) É necessário dispender tempo e esforço para divulgar o Sistema, apresentar seus benefícios e motivar o uso, pois os pesquisadores possuem maneiras muito próprias de trabalho e já estão habituados com certas ferramentas.

8. DESENVOLVIMENTOS FUTUROS

Conforme avaliação da equipe de analistas de negócio e dos primeiros usuários do aplicativo, apesar dos bons resultados obtidos até o momento, esta Pesquisa não pode ser considerada concluída. Dentre os possíveis trabalhos futuros legados ao informata, destacam-se:

a) Aprofundar seus conhecimentos quanto à biologia molecular computacional envolvida nos projetos genoma da Rede Biofoco para continuar propondo melhorias ao Sistema.

b) Em sintonia com o bioinformata, deverá continuar a pesquisa de novas análises a serem incluídas na plataforma do Sistema BLOOM. As primeiras análises candidatas são: enviar seqüências para modelagem estrutural no *Modeller* [lvii] e previsão de domínios de proteínas.

c) Identificar as possibilidades de reuso dos atuais componentes gráficos, tanto para o Sistema Genoma quanto para o Sistema Anotação. As primeiras iniciativas serão: melhorar a visualização gráfica da montagem (*assembling*) de reads, que devem formar contigs e gerar fragmentos que correspondem fisicamente ao cromossomo; e localização de COGs[89] no Sistema Anotação. Também deverá haver estudo para emprego dos componentes no mapeamento do genoma estrutural.

d) Para a segunda fase do Sistema BLOOM, considerar o emprego de componentes da plataforma J2EE. Esses componentes não foram utilizados na primeira fase devido à inexistência da Persistência Gerenciada pelo Contêiner (Container-Managed Persistence – CMP), a qual poderia substituir o serviço de persistência de CORBA. Outro ponto importante quanto a não aplicação imediata de J2EE é que ainda não se vislumbra um padrão estável entre as linguagens de consulta EJBQL[lviii] e JDOQL[lix].

e) Implementar maneiras úteis de disponibilizar os resultados das análises. Duas possibilidades iniciais serão: relatório impresso ou arquivo de dados em formato texto e PDF.

[lvii] Modeller – vide http://salilab.org/modeller/modeller.html.

[lviii] EJBQL – *Enterprise Java Bean Query Language* ou Linguagem de Consulta para EJB. Vide http://www.javaworld.com/javaworld/jw-03-2002/jw-0301-dao.html.

[lix] JDOQL – *Java Data Object Query Language* ou Linguagem de Consulta para Objetos Java de Dados. Vide http://www.fawcette.com/javapro/2002_07/magazine/features/djordan/default.asp.

f) Melhorar performance do Sistema pela utilização de XML para persistência de dados dos Serviços de Parametrização e de Filtragem. Esses atributos estão sendo mantidos atualmente em tabelas relacionais. Os arquivos XML serão compactados no .jar da aplicação e, ao fazer o *download* do Sistema, o usuário terá acesso local a esses dados .

g) Estudar detalhadamente os diversos formatos de seqüências para implementar o *façade* de cada ferramenta externa a ser integrada à plataforma do Sistema.

h) Utilizar o recurso de Internationalization da linguagem Java para disponibilizar versões do Sistema também em inglês e espanhol.

i) Aprofundar o estudo dos componentes do kit BlastView e da biblioteca BioJava.

9. REFERÊNCIAS BIBLIOGRÁFICAS

[Allamaraju, 2000] Allamaraju, Subrahmanyam et al. "*Professional Java Server Programming J2EE Edition* ". Computer Science. Wrox Press Ltd, 2000.

[Altschul e outros, 1997] Altschul, Stephen F.; Madden, Thomas L.; Schäffer, Alejandro; Zhang, A. Jinghui; Zhang, Zheng; Miller, Webb e Lipman, David J. "*Gapped BLAST and PSI-BLAST: a new generation of protein database search programs"*. 1997. http://www.cs.cornell.edu/Courses/CS726/2001fa/presentations/niranjan2.ppt. *Site* pesquisado em 12/03/2002.

[Araújo, 2002] Araújo, Alan Henriques de. "*Thomas Malthus e a explosão demográfica*". Economista, analista militar e financeiro editor do site U.S NAVY na NET. http://www.alanhenriques1.hpg.ig.com.br/artigoeconomia03.html . *Site* pesquisado em 02/12/2002.

[Baxevanis e Ouellette, 2001] Baxevanis, Andreas D.; Ouellette, B. F. Francis. "*Bioinformatics: A Pratical Guide to the Analysis of Genes and Proteins*". Wiley-Interscience. USA, 2001.

[Bioinfo_326, 2002] Universidade Municipal de Nova York **(CUNY).** "*Introduction to Bioinformatics and Genomics*", maio de 2002. Departamento de Biologia. http://www.library.csi.cuny.edu/~davis/Bioinfo_326/lectures/ . *Site* pesquisado em 25/11/2002.

[Bionavigator, 2002] Bionavigator – http://www.bionavigator.com. Site pesquisado em agosto de 2002.

[BLC, 1999] Centro de Aprendizado de Biologia (BLC). "*Bioinformatics and Genomic Analysis*", janeiro de 1999. Universidade do Arizona, USA. http://www.blc.arizona.edu/courses/bioinformatics/. *Site* pesquisado em 01/03/2002.

[BLOCKS, 2001] BLOCKS. "*Blocks WWW Server*", março de 2002. http://blocks.fhcrc.org/blocks/. *Site* pesquisado em 25/03/2002.

[Bonato e Bonato, 2001] Bonato, Chris; Bonato, Cezar. "*Biologia na Web*". Professora de Genética e professor de Física da UFPB. Paraíba, Brasil. http://www.biologianaweb.com/. *Site* pesquisado em 03/10/2001.

[Booch et al, 2000] Grady Booch, James Rumbaugh e Ivar Jacobson. "*UML, Guia do Usuário* ". Rio de Janeiro : Campus, 2000.

[Brodkin, 2001] Brodkin, Sam. "*Use XML data binding to do your laundry*". http://www.javaworld.com/javaworld/jw-12-2001/jw-1228-jaxb.html. *Site* pesquisado em 03/06/2002.

[Cooper, 2002] Cooper, James W. *"The Design Patterns Java Companion"*. http://www.patterndepot.com/put/8/JavaPatterns.htm. Site pesquisado em 25/06/2002.

[Craven, 2002] Craven, Mark. *"Heuristic Methods for Sequence Database Searching"*, fevereiro de 2002. Email craven@biostat.wisc.edu. http://www.biostat.wisc.edu/~craven/776/lecture4.pdf. *Site* pesquisado em 05/04/2002.

[Developers, 2001] *ASPECTOS DO DESENVOLVIMENTO MULTIPLATAFORMA*. Publicado originalmente na Developers' Magazine (www.developers.com.br), número 59, julho/2001, pg. 18-20. http://www.geocities.com/acfconsult/artigo5.htm. Site pesquisado em 25/06/2002.

[Dipperstein, 2000] Dipperstein, Michael. "*DNA Sequence Database*", outubro de 2000. http://www.cs.ucsb.edu/~mdipper/dna/DNApaper.html . *Site* pesquisado em 25/03/2002.

[FAPESP, 1999] Pesquisa FAPESP (Fundação de Amparo à Pesquisa do Estado de São Paulo). "*Projeto Genoma*", outubro de 1999. http://www.fapesp.br/ . *Site* pesquisado em 20/11/2002.

[Ferlanti et al, 1999] Ferlanti, Erik S.; Ryan, Joseph F.; Makalowska, Izabela; e Baxevanis, Andreas D. "*WebBLAST 2.0: an integrated solution for organizing and analyzing sequence data*". Bioinformatics Applications Note, Volume 15, Número 5, Páginas 422-423, USA, 1999.

[Fernandes-Matioli, 2001] Fernandes-Matioli, Flora Maria de Campos. "*VIII Simpósio de Citogenética e Genética de Peixes*". http://adi-38.bio.ib.usp.br/sismo/index.html . *Site* pesquisado em 25/03/2002.

[Finch, 2002] Finch®-Suite **-** http://www.geospiza.com/products/finch-suite/index.htm. Site pesquisado em agosto de 2002.

[GENTROP, 2001] GENTROP - Grupo de Pesquisa em Genética Molecular e Bioquímica de Doenças Tropicais. "*Genética Molecular: fundamento e aplicações*". Departamento de Genética Universidade Federal de Pernambuco. Recife, PE. Brasil. http://www.progene.ufpe.br/Apostilas/GENMOL1.htm. *Site* pesquisado em 14/09/2001.

[Gusfield, 1997] Gusfield, Dan. "*Algoritms on Strings, Trees and Sequences*". Computer Science and Computational Biology. Cambridge University Press, 1997.

[Heath e Ramakrishnan, 2002] Heath, Lenwood S. e Ramakrishnan, Naren. "*The Emerging Landscape of Bioinformatics Software Systems*". Computer, Volume 35, Número 7, USA, julho de 2002.

[Hunter, 1993] Hunter, Lawrence. "*Artificial Intelligence and Molecular Biology*". MIT Press, 1993.

[InterPro, 2001] InterPro. "*TheInterPro Database*", março de 2002. http://www.ebi.ac.uk/interpro/index.html. *Site* pesquisado em 25/03/2002.

[J2EE, 2002] Designing Enterprise Applications with the J2EE™ Platform, segunda edição. http://java.sun.com/blueprints/guidelines/designing_enterprise_applications_2e/introduction/introduction.html#1045529 . Site pesquisado em julho de 2002.

[JalView, 2002] JalView - Java multiple alignment editor - http://www.es.embnet.org/Doc/jalview/contents.html e http://www2.ebi.ac.uk/~michele/jalview/contents.html. Sites pesquisados em agosto de 2002.

[Kim, 2002] Kim, Junhyong. "*Computers are from Mars, Organisms are from Venus*". Computer, Volume 35, Número 7, USA, julho de 2002.

[Kuhn, 2001] Kuhn, Gustavo C.S. "*Multiple Sequence Alignment*" Email gcskuhn@rge.fmrp.usp.br. http://www.rge.fmrp.usp.br/cursos/topicosiii/paginas/web20/20multiple.htm. *Site* pesquisado em 05/05/2002.

[Le Novère, 2002] Le Novère, Nicolas. "*Protein secondary structure prediction*". Instituto Pasteur. França, 2002. http://www.pasteur.fr/recherche/unites/neubiomol/secstrpr.html. *Site* pesquisado em 26/04/2002.

[Mangan, 1999] Mangan, Marco Aurélio Souza; Vargas, Patrícia Kayser; Azzolin, Denny. *"Técnicas para Desenvolvimento de Aplicações Orientadas a Objetos Utilizando a Linguagem Java"*. http://www.inf.ufrgs.br/~kayser/sblp99/TutorialSBLP99.html. Site pesquisado em 25/06/2002.

[Martins, 2000] Martins, Wellington S. . "*Discovery Informatics I (Bioinformatics)*". Prof. Dr. do Laboratório de Sistemas Paralelos e Arquitetura de Computadores, Universidade de Delaware, USA . ftp://ftp.capsl.udel.edu/pub/courses/eleg667/2000/slides/ (tópicos 2a, 2b e 2c). *Site* pesquisado em 26/04/2002.

[Meidanis e Setubal, 1997] Meidanis, João; Setubal, João "*A introduction to Computational Molecular Biology*". International Thomson Publishing. USA, 1997. (meidanis@dcc.unicamp.br e setubal@dcc.unicamp.br).

[Mitchell, 1996] Mitchell, Melanie. "*A introduction to genetic algorithms*". MIT Press, 1996. (mm@santafe.edu)

[NCBI/Education, 2002] Centro Nacional de Informação Biotecnológica (NCBI). "*Education*", março de 2002. http://www.ncbi.nlm.nih.gov/Education/index.html. *Site* pesquisado em 25/03/2002.

[NCBI/Primer, 2002] Centro Nacional de Informação Biotecnológica (NCBI). "*A Science Primer*", janeiro de 2002. http://www.ncbi.nih.gov/About/primer/index.html . *Site* pesquisado em 25/03/2002.

[Orfali et al, 1997] Orfali, Robert; Harkey, Dan; Edwards, Jeri. "*Instant CORBA*". John Wiley & Sons, 1997.

[Pappas, 2002] Pappas, Georgios Joannis J. . "*Rede de pesquisa e desenvolvimento em bioinformática do Centro-Oeste* ". (gpappas@pos.ucb.br)

[Pearson, 2001] Pearson, William R. "*Protein Sequence comparison and Protein Evolution*". Tutorial – ISMB2000. Department of Biochemistry and Molecular Genetics. University of Virginia, USA. October, 2001. (wrp@virginia.EDU)

[Pearson, 2002] Pearson, William R. *"FASTA : Sequence database search (version 3)"*, março de 2002. http://bioweb.pasteur.fr/seqanal/interfaces/fasta.html . *Site* pesquisado em 05/04/2002.

[Pereira, 2001] Pereira, Gonçalo Amarante Guimarães. "*O projeto genoma de Crinipellis perniciosa, fungo causador da vassoura de bruxa: fundamentos e potencial contribuição*". Departamento de Genética e Evolução. Instituto de Biologia. UNICAMP, Brasil. (goncalo@unicamp.br)

[PROSITE, 2002] PROSITE. "*Banco de dados de famílias e domínios de proteínas*", março de 2002. http://www.expasy.ch/prosite/. *Site* pesquisado em 25/03/2002.

[PSC, 1999] Centro de Supercomputação de Pittsburgh (PSC). "*Sequnce Analysis: Which scoring method should I use?*", novembro de 1999. USA. http://www.psc.edu/research/biomed/homologous/scoring_primer.html. *Site* pesquisado em 01/03/2002.

[Rocha, 2000] Rocha, Eduardo. "*Folhas de Bioinformática e Análise de Seqüências*". Instituto Pasteur. França, 2000. (erocha@pasteur.fr) http://wwwabi.snv.jussieu.fr/people/erocha/curso.pt.pdf. *ite* pesquisado em 03/10/2001.

[Raupp, 2002] Raupp, Fernanda. "*Similaridades, alinhamentos e perfis*". http://www.lncc.br/~biologia/downloads/sapdownload.pdf. S*ite* pesquisado em 03/04/2002.

[Russell, 2002] Russell, Rob. "*A Guide to Structure Prediction (version 2.1)"*. Alemanha, 2001. http://speedy.embl-heidelberg.de/gtsp/secstrucpred.html. S*ite* pesquisado em 03/04/2002.

[Santos, 2000] Santos, Francisco Prosdocimi. "*Filogenia molecular"*. Brasil, 2000. http://luna.spaceports.com/~franc/MolecEvol/evolucao_molecular.htm. S*ite* pesquisado em 03/04/2002.

[Santos, 2002] Santos, Fabrício R. . "*Evolução molecular"*. Brasil, 2002. http://www.icb.ufmg.br/~lbem/aulas/grad/evol/evolmol/. S*ite* pesquisado em 03/04/2002.

[Santos e Ortega, 2001] Santos, Fabrício R.; Ortega, José Miguel. "*Bioinformática aplicada à Genômica"*. Brasil, 2002. www.icb.ufmg.br/~lbem/aulas/grad/tge/bioinfo/bioinfogenomica.pdf . S*ite* pesquisado em 03/10/2002.

[Sipert, 1998] Sipert, Lisiane Volpi; Martins, Vidal. "*DESIGN PATTERNS – COMPOSITE"*. http://www.pr.gov.br/celepar/celepar/batebyte/edicoes/1998/bb75/design.htm. Site pesquisado em 25/06/2002.

[Struts, 2002] Jakarta-Struts - http://jakarta.apache.org/struts/resources/. Site pesquisado em agosto de 2002.

[UFRJ, 2002] "*As Bases Moleculares da Herança Genética"*. http://www.iq.ufrj.br/~joab/iqb201/tutorial/dna/historia/historico-dna.html. Site pesquisado em 03/05/ 2002.

[WebBlast, 2002] WebBlast – http://research.nhgri.nih.gov/apps/webblast e http://research.nhgri.nih.gov/software/webblast2/bioinformatics/webblast_article.shtml. Sites pesquisados em agosto de 2002.

Outras referências (pesquisadas no período de 03/10/2001 a 20/01/2003) :

http://bioinfo.weizmann.ac.il/courses/BCG/lectures/02_*pairwise*/2.3quantitative/02distance.html
http://bioinformatics.utmem.edu/classes/module9/overview.html
http://ca.expasy.org/sprot/relnotes/relstat.html
http://ca.expasy.org/sprot/sprot_details.html.
http://cogprints.ecs.soton.ac.uk/bbs/Archive/bbs.pinker.html
http://esg-www.mit.edu:8001/esgbio/lm/proteins/structure/structure.html
http://gened.emc.maricopa.edu/bio/bio181/BIOBK/BioBookPROTSYn.html
http://ismb00.sdsc.edu/tutorials/samudrala.html
http://lectures.molgen.mpg.de/
http://library.thinkquest.org/19926/text/tour/18.htm
http://micro.magnet.fsu.edu/aminoacids/index.html
http://piopio.school.nz/revin.htm
http://pir.georgetown.edu/cgi-bin/iproclass/nfhist.pl?n=all
http://research.nhgri.nih.gov/software/webblast2/bioinformatics/webblast_article.shtml
http://www.accessexcellence.org/AB/GG/central.html
http://www.biologianaweb.com/genomica/buscas.html
http://www.biologianaweb.com/Livro2/C4/moldna.html
http://www.biologianaweb.com/navegando/prepos.html
http://www.blc.arizona.edu/courses/bioinformatics/blosum.html
http://www.blc.arizona.edu/courses/bioinformatics/dayhoff.html
http://www.cbs.dtu.dk/dave/roanoke/genetics980406.htm
http://www.cosmobrain.com/cosmobras/res/grego.html#top
http://www.cs.ualberta.ca/~charter/seqcmp-notes.html
http://www.cut-the-knot.com/do_you_know/Strings.html
http://www.ddbj.nig.ac.jp/ddbjnew/relinfo-e.html
http://www.dkfz-heidelberg.de/tbi/bioinfo/Biol/index.html
http://www.ebi.ac.uk/embl/News/news.html

http://www.fao.org/sd/RTdirect/RTre0036.htm#topofpage
http://www.fapesp.br/ciencia47.htm
http://www.fapesp.br/genoma391.htm
http://www.gdb.org/
http://www.genetics.wustl.edu/bio5491/blast.html
http://www.genomicglossaries.com/content/lifesciences_databasesdirectory.asp
http://www.geocities.com/sunsetstrip/club/2443/aaprot.htm
http://www.library.csi.cuny.edu/%7Edavis/Bioinfo_326/bioinfo_glossary.html
http://www.lncc.br/~biologia/home.html
http://www.math.tau.ac.il/~rshamir/algmb/00/scribe00/html/lec02/node4.html
http://www.mblab.gla.ac.uk/dictionary/ (dicionário de biologia molecular)
http://www.mips.biochem.mpg.de/proj/protseqdb/
http://www.ncbi.nih.gov/About/primer/bioinformatics.html
http://www.ncbi.nih.gov/About/primer/index.html (*)
http://www.ncbi.nih.gov/Education/BLASTinfo/glossary2.html
http://www.ncbi.nih.gov/Education/BLASTinfo/Orthology.html
http://www.ncbi.nih.gov/Education/BLASTinfo/Scoring2.html
http://www.ncbi.nlm.nih.gov/BLAST/tutorial/Altschul-1.html (statistics)
http://www.ncbi.nlm.nih.gov/Entrez/
http://www.ncbi.nlm.nih.gov/Genbank/genbankstats.html.
http://www.nhgri.nih.gov/DIR/VIP/Glossary/pub_glossary.cgi
http://www.ornl.gov/hgmis/publicat/primer/prim2.html
http://www.ornl.gov/TechResources/Human_Genome/glossary/glossary_s.html
http://www.progene.ufpe.br/Apostilas/GENMOL1.htm
http://www.rcsb.org/pdb/holdings.html
http://www.tmbioscience.com/glossary.jsp
http://www.umanitoba.ca/faculties/afs/plant_science/courses/39_769/
http://www.zoology.ubc.ca/~otto/EvolDisc/Glossary.html
http://www3.ebi.ac.uk/Services/DBStats/
http://www.virtual.epm.br/cursos/biomol/biom.htm (Escola Paulista de Medicina)

ANEXO

Em anexo, seguem: um exemplo de resultado de BLAST no formato XML; um diagrama de classes de controle comuns aos Sistemas Genoma e BLOOM; e as 5 primeiras telas de acesso ao Sistema BLOOM, passando pelo Sistema Genoma.

Iniciando-se nesta página e se estendendo pelas próximas duas páginas, segue um exemplo de resultado de BLASTn no formato XML para uma seqüência de nucleotídeos (Figura 0-1). O resultado completo preencheu 69 páginas no formato Word Windows:

```
<?xml version="1.0" ?>
<!DOCTYPE BlastOutput (View Source for full doctype...)>
 - <BlastOutput>
      <BlastOutput_program>blastn</BlastOutput_program>
      <BlastOutput_version>blastn 2.2.4 [Aug-26-2002]</BlastOutput_version>
      <BlastOutput_reference>~Reference: Altschul, Stephen F., Thomas L. Madden, Alejandro A.
        Schaffer, ~Jinghui Zhang, Zheng Zhang, Webb Miller, and David J. Lipman (1997), ~"Gapped
        BLAST and PSI-BLAST: a new generation of protein database search~programs", Nucleic
        Acids Res. 25:3389-3402.</BlastOutput_reference>
      <BlastOutput_db>nr</BlastOutput_db>
      <BlastOutput_query-ID>lcl|1_2381</BlastOutput_query-ID>
      <BlastOutput_query-def />
      <BlastOutput_query-len>446</BlastOutput_query-len>
    - <BlastOutput_param>
       - <Parameters>
            <Parameters_expect>10</Parameters_expect>
            <Parameters_sc-match>1</Parameters_sc-match>
            <Parameters_sc-mismatch>-3</Parameters_sc-mismatch>
            <Parameters_gap-open>5</Parameters_gap-open>
            <Parameters_gap-extend>2</Parameters_gap-extend>
            <Parameters_filter>L;</Parameters_filter>
      </Parameters>
   </BlastOutput_param>
    - <BlastOutput_iterations>
       - <Iteration>
            <Iteration_iter-num>1</Iteration_iter-num>
          - <Iteration_hits>
             - <Hit>
                  <Hit_num>1</Hit_num>
                  <Hit_id>gi|22477517|gb|BC037097.1|</Hit_id>
                  <Hit_def>Mus musculus, clone MGC:46891 IMAGE:5345693, mRNA, complete
                    cds</Hit_def>
                  <Hit_accession>BC037097</Hit_accession>
                  <Hit_len>2732</Hit_len>
                - <Hit_hsps>
                   - <Hsp>
                        <Hsp_num>1</Hsp_num>
                        <Hsp_bit-score>811.277</Hsp_bit-score>
                        <Hsp_score>409</Hsp_score>
                        <Hsp_evalue>0</Hsp_evalue>
                        <Hsp_query-from>34</Hsp_query-from>
                        <Hsp_query-to>446</Hsp_query-to>
                        <Hsp_hit-from>2077</Hsp_hit-from>
```

```
<Hsp_hit-to>2489</Hsp_hit-to>
<Hsp_query-frame>1</Hsp_query-frame>
<Hsp_hit-frame>1</Hsp_hit-frame>
<Hsp_identity>412</Hsp_identity>
<Hsp_positive>412</Hsp_positive>
<Hsp_align-len>413</Hsp_align-len>

<Hsp_qseq>GCAGAGACCGTCGGCCGTGAGGTGTTTGCAGCATATCACTCTG
CTGTAATCAGTGTGTCGCTTCTGCACAATCAGAGACTGTCTCATCTCTCCACT
CAACGTGGAAGTTGCCTTGTGCCTAAACTGAATTGACAAATGCATTGTAACT
ACAAATTTTATTTATTGTTATGGAACTGTGAGGTCTACATATAAAGGGAAAAG
TTCATGTGGGAAGCTGGTGTACACTCAGCTGATGCCAGCATTGTTAAAGCTG
TTCACAGAGCAGTGGCAACCATTGGCCCTTAGCATTCCCGGCATACCTGTTA
GTGTCTTAAAAAGGAAGGGAGTCCTTTGTTGCCCTCTCCGACCTTCGCCATA
TGAATAGTGATTTCCATGAAATAGGAAAAATATTACTTCGTATAGCATTTCTC
TCT</Hsp_qseq>

<Hsp_hseq>GCAGAGACCGTCGGCCGTGAGGTGTTTGCAGCATATCACTCTG
CTGTAATCAGTGTGTCGCTTCTGCACAATCAGAGACTGTCTCATCTCTCCACT
CAACGTGGAAGTTGCCTTGTGCCTAAACTGAATTGACAAATGCATTGTAACT
ACAAATTTTATTTATTGTTATGGAACTGTGAGGTCTACATATAAAGGGAAAAG
TTCATGTGGGAAGCTGATGTACACTCAGCTGATGCCAGCATTGTTAAAGCTG
TTCACAGAGCAGTGGCAACCATTGGCCCTTAGCATTCCCGGCATACCTGTTA
GTGTCTTAAAAAGGAAGGGAGTCCTTTGTTGCCCTCTCCGACCTTCGCCATA
TGAATAGTGATTTCCATGAAATAGGAAAAATATTACTTCGTATAGCATTTCTC
TCT</Hsp_hseq>

<Hsp_midline>|||||||
```

```
                    AACGTGGAAGTTGCCTTGTGCCTAAACTGAATTGACAAATGCATTGTAACTA
                    CAAATTTTATTTATTGTTATGGAACTGTGAGGTCTACATATAAAGGGAAAAGT
                    TCATGTGGGAAGCTGGTGTACACTCAGCTGATGCCAGCATTGTTAAAGCTGT
                    TCACAGAGCAGTGGCAACCATTGGCCCTTAGCATTCCCGGCATACCTGTTAG
                    TGTCTTAAAAAGGAAGGGAGTCCTTTGTTGCCCTCTCCGACCTTCGCCATAT
                    GAATAGTGATTTCCATGAAATAGGAAAAATATTACTTCGTATAGCATTTCTCT
                    CT</Hsp_qseq>
                  <Hsp_hseq>GCAGAGACCGTCGGCCGTGAGGTGTTTGCAGCATATCACTCTGCT
                    GTAATCAGTGTGTCGCTTCTGCACAATCAGAGACTGTCTCATCTCTC-
                    ACTCAACGTGGAAGTTGCCTTGTGCCTAAACTGAATTGACAAATGCATTGTA
                    ACTACAAATTTTATTTATTGTTATGGAACTGTGAGGTCTACATATAAAGGGAA
                    AAGTTCATGTGGGAAGCTGATGTACACTCAGCTGATGCCAGCATTGTTAAAG
                    CTGTTCACAGAGCAGTGGCAACCATTGGCCCTTACGATTCCCGGCATACCTG
                    TTAGTGTCTTAAAAAGGAAGGGAGTCCTTTGTTGCCCTCTCCGACCTTCGCC
                    ATATGAATAGTGATTTCCATGAAATAGGAAAAATATTACTTCGTATAGCATTT
                    CTCTCT</Hsp_hseq>
                  <Hsp_midline>|||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||
                    ||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||
                    ||||||||||||||||||||||||||||||||||||||||||||||||||||||||
                    ||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||</Hsp_midline>
                </Hsp>
              </Hit_hsps>
            </Hit>
          ……...
          ……...
          ……Outros hits……...
          ……...
          ……...
        </Iteration_hits>
        - <Iteration_stat>
          - <Statistics>
              <Statistics_db-num>968826</Statistics_db-num>
              <Statistics_db-len>-137679</Statistics_db-len>
              <Statistics_hsp-len>0</Statistics_hsp-len>
              <Statistics_eff-space>0</Statistics_eff-space>
              <Statistics_kappa>0.710605</Statistics_kappa>
              <Statistics_lambda>1.37407</Statistics_lambda>
              <Statistics_entropy>4.94066e-324</Statistics_entropy>
            </Statistics>
          </Iteration_stat>
        </Iteration>
      </BlastOutput_iterations>
    </BlastOutput>
```

Figura 0-1 – Um exemplo de resultado BLAST no formato XML.

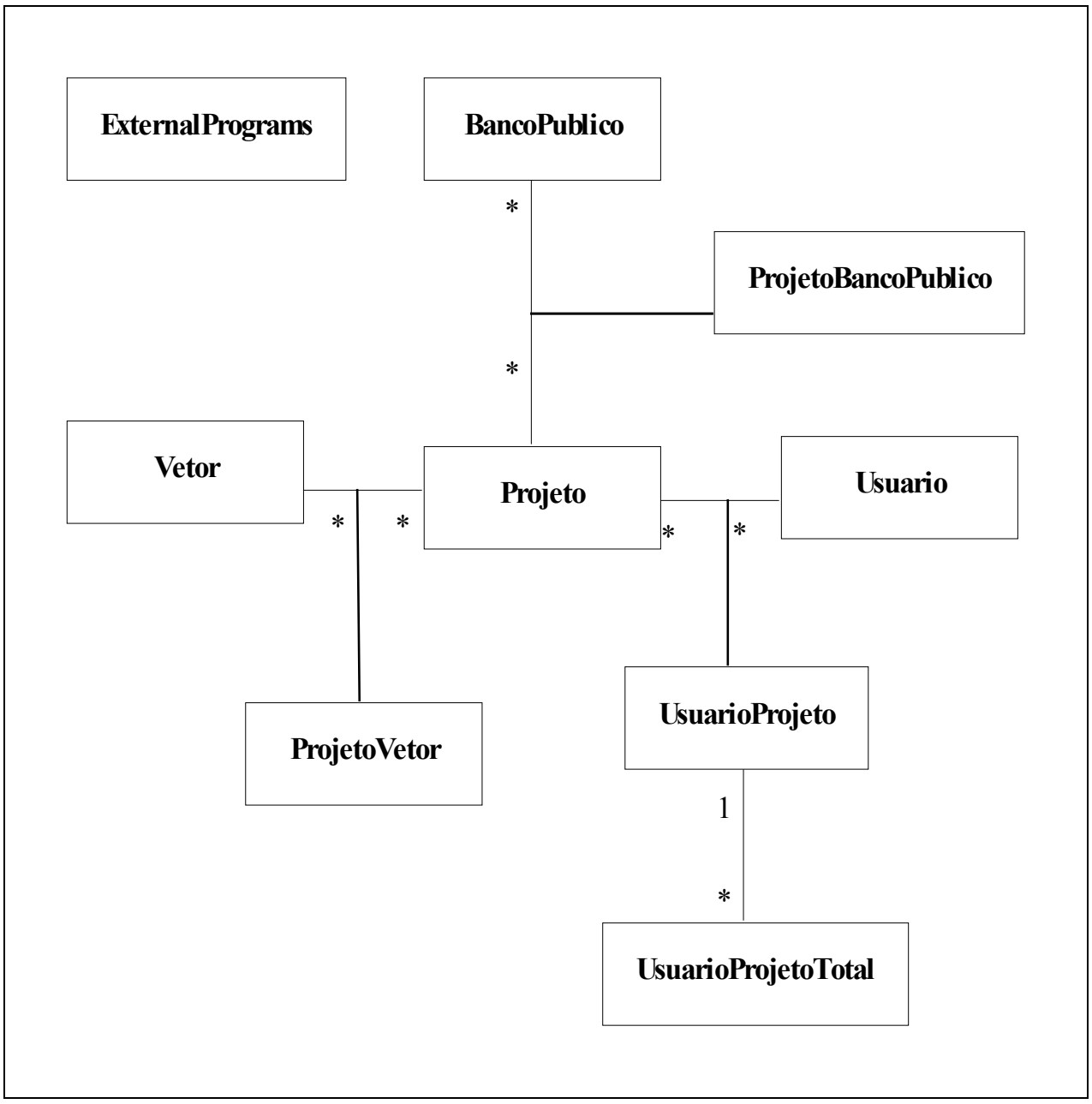

Figura 0-2 – Sistemas Genoma e BLOOM: diagrama de classes de controle comuns a ambos.

A seguir, as 5 primeiras telas de acesso ao Sistema BLOOM, passando pelo Sistema Genoma:

Figura 0-1 – Sistema Genoma: validação do usuário e entrada.

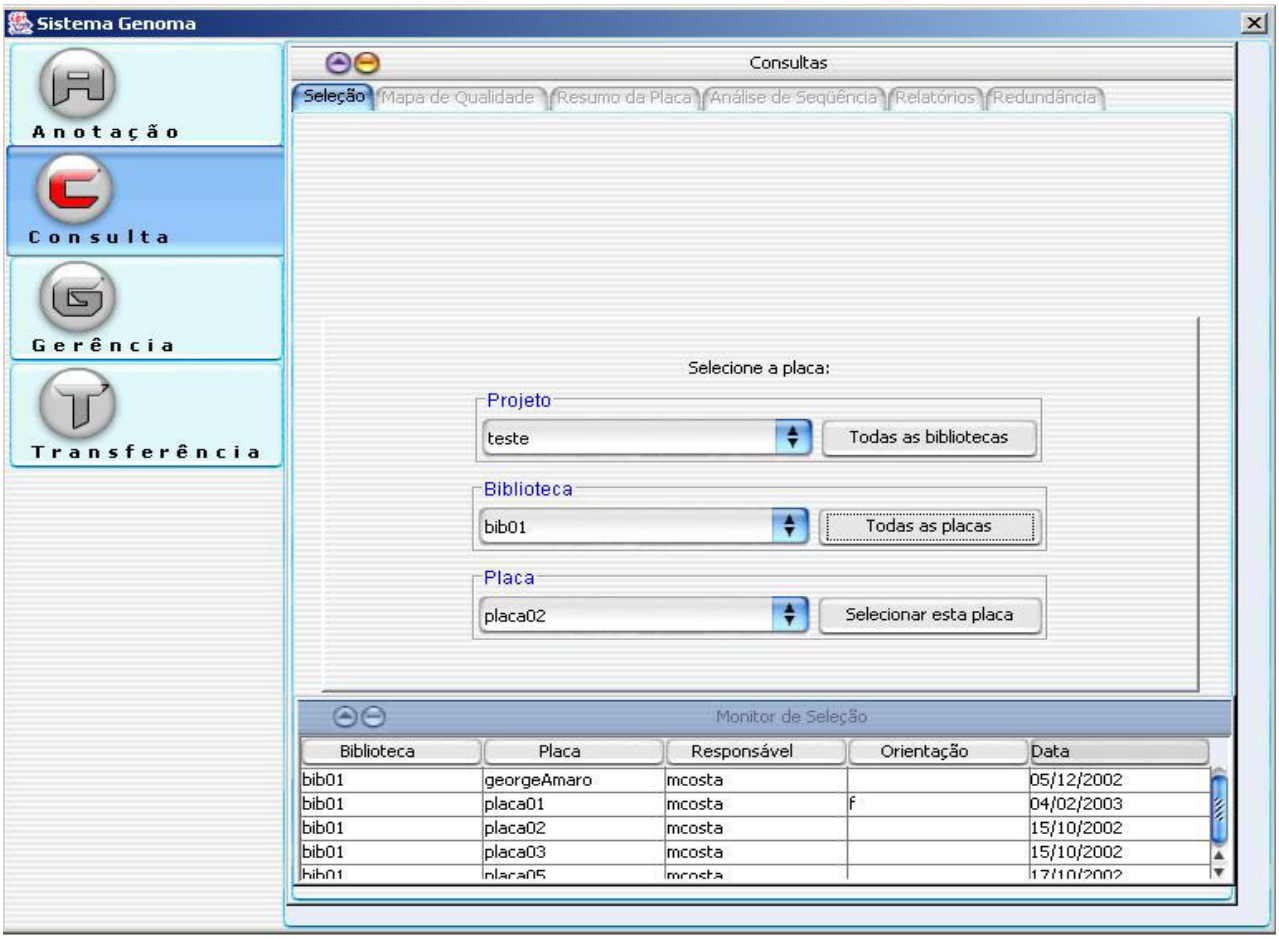

Figura 0-2 - Sistema Genoma: módulo de Consulta – Seleção da placa.

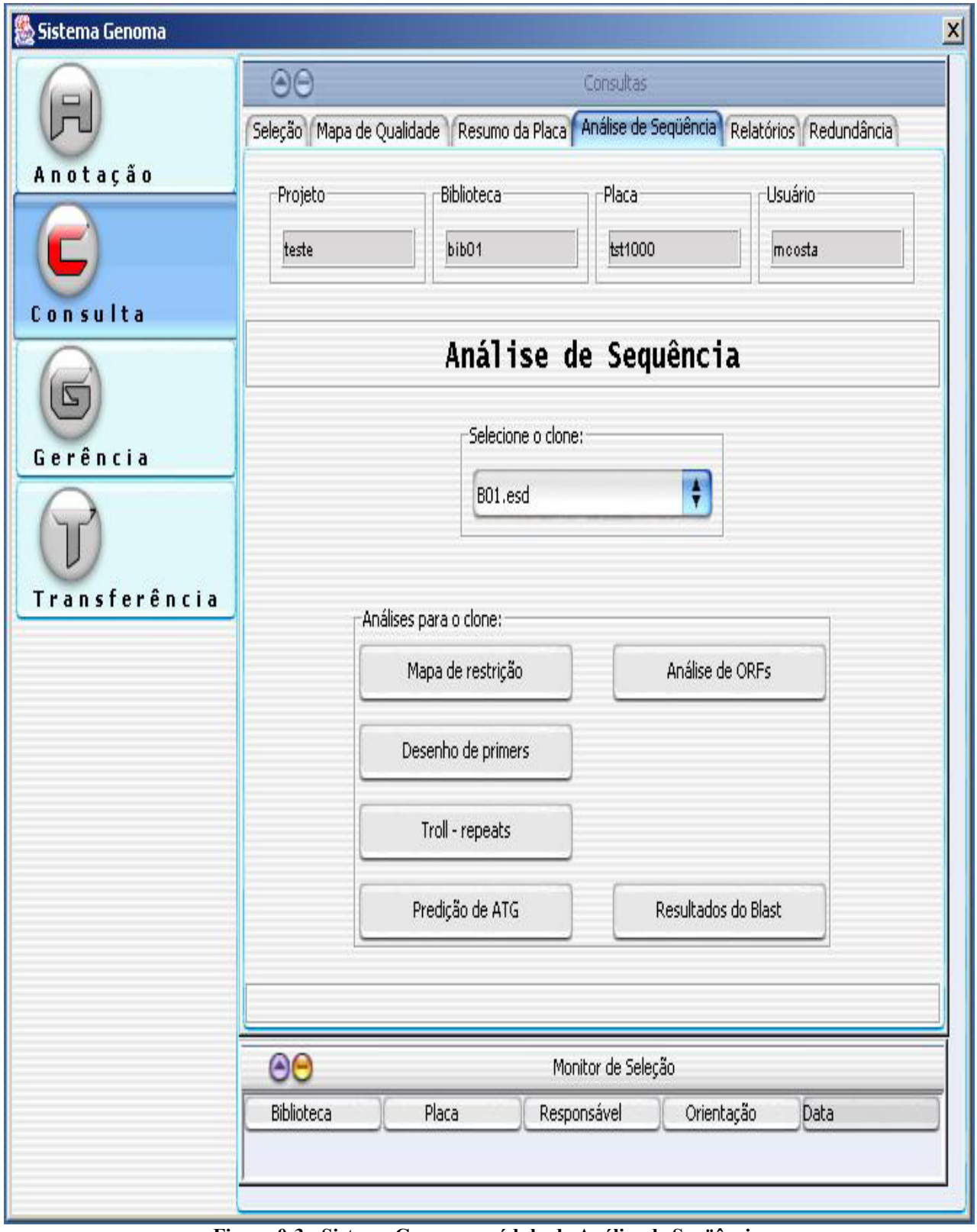

Figura 0-3 - Sistema Genoma: módulo de Análise de Seqüência.

Para entrar no Sistema BLOOM, clicar no botão Resultados do BLAST.

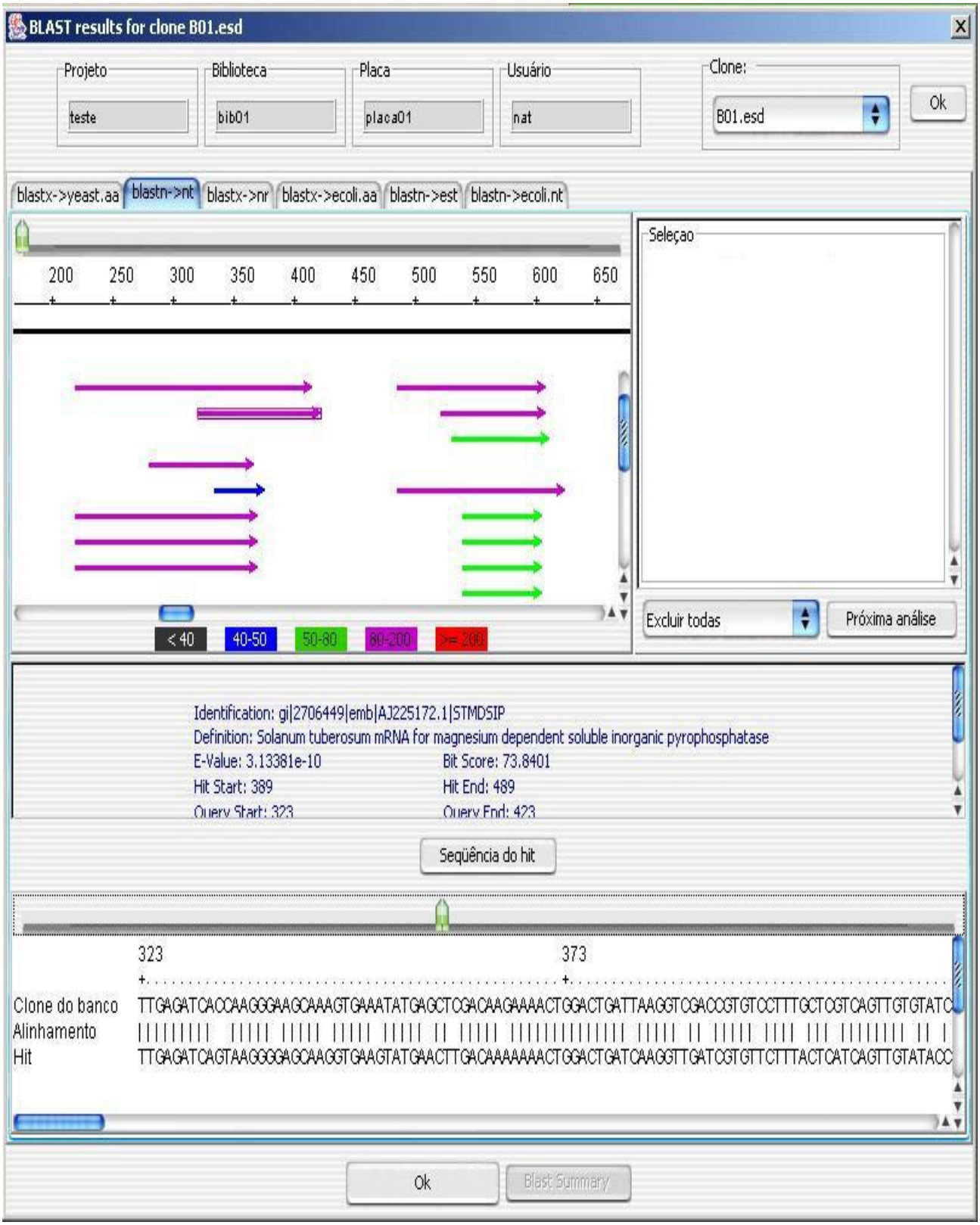

Figura 0-4 – Sistema BLOOM: visualização gráfica do resultado do BLAST.

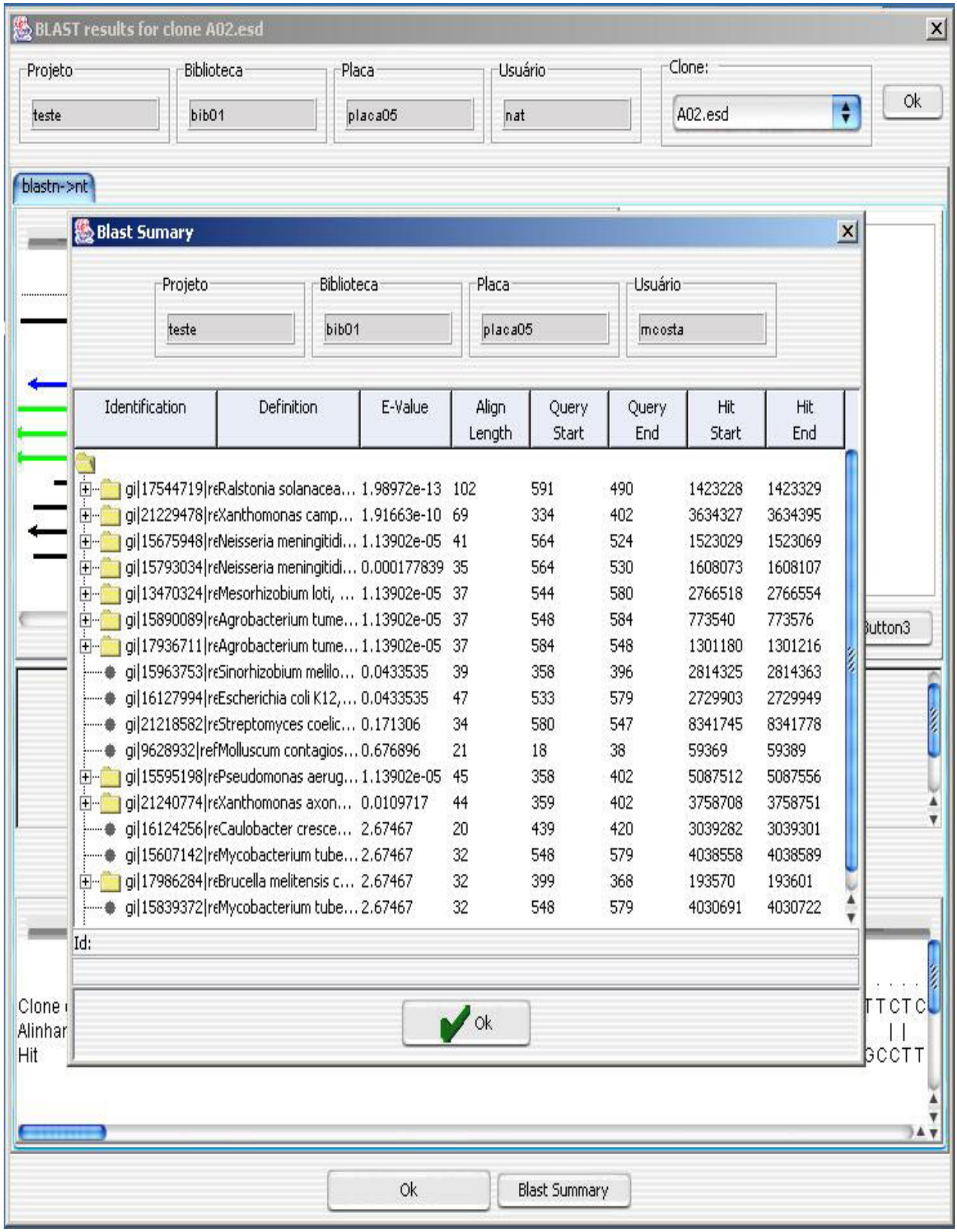

Figura 0-1 – Sistema BLOOM: sumário do resultado do BLAST.

GLOSSÁRIO

[1] **Lei de Moore** - foi criada há 30 anos por Gordon Moore, fundador da Intel, e estabelece que a capacidade dos chips é duplicada no período de 18 a 24 meses.

[2] **Pares de bases** – bp ou *base pair* em inglês, diz-se pares de bases para expressar que, na hélice (ou fita) dupla do DNA, a base (A, C, G e T) de uma das fitas possui uma base complementar na outra fita. Pode-se também dizer simplesmente bases.

[3] ***Drosophila melanogaster*** – ou mosca da fruta, é um pequeno inseto de 3mm de comprimento e que se acumula ao redor de uma fruta estragada. É um dos mais valiosos organismos em pesquisa biológica, particularmente em genética e biologia do desenvolvimento. Suas principais características são: tamanho reduzido; manuseio fácil e barato de grandes quantidades; genoma pequeno (~ 165 Mb) já seqüenciado em 2001; rápido ciclo de vida (~2 semanas); disponibilidade de tipos mutantes; pesquisada mundialmente por governos, laboratórios e universidades. Vide http://ceolas.org/VL/fly/intro.html.

[4] ***Arabidopsis thaliana*** – é uma pequena planta de florescência usada largamente como um organismo modelo em genética e biologia molecular de plantas, embora não seja de grande significância agronômica. É membro da família da mustarda (Brassicaceae) , que inclue espécies cultivadas, tais como o repolho e o rabanete. Suas principais características são: genoma pequeno (~ 125 Mb) já seqüenciado no ano de 2000; rápido ciclo de vida (~6 semanas da germinação à maturação das sementes); alta produtividade de sementes; disponibilidade de linhas mutantes; pesquisada mundialmente por governos; laboratórios e universidades. Vide www.arabidopsis.org.

[5] **Modelagem de estruturas** – é a reconstrução tridimensional da forma da proteína por meio de sistemas de Visão Computacional, tendo como entrada as projeções bidimensionais obtidas por microscopia. Vide http://www.dcc.ufmg.br/pos/html/spg97/anais/html/omni.html.

[6] **Hits** – seqüências coincidentes ou similares àquela informada como parâmetro de busca. O termo é mantido em inglês para compatibilizar com o vocabulário freqüentemente empregado pelos pesquisadores em bioinformática no Brasil.

[7] **Escore em bits** – ou *Bit score* em inglês, é o valor do escore S normalizado para S', pela fórmula S' = (lambda*S - ln K)/(ln 2), sendo que lambda e K dependem do sistema de escore (matriz e gaps). Vide http://www.ncbi.nlm.nih.gov/Education/BLASTinfo/glossary2.html.

[8] **E-value** – ou *Expectation value* em inglês, é o número de diferentes alinhamentos com escores eqüivalentes ou melhores que o escore S e, conforme esperado, poderão ocorrer por acaso em buscas no banco de dados. Por isso, quanto menor o E-value, mais significante o escore. Vide http://www.ncbi.nlm.nih.gov/Education/BLASTinfo/glossary2.html.

[9] **P-value** – ou *Probability value* em inglês, é a probabilidade de um alinhamento ocorrer com o mesmo escore em questão ou melhor. O p-value é calculado pelo valor do escore S relativo à distribuição esperada dos escores dos HSPs de comparações com seqüências aleatórias de mesmo tamanho e mesma composição como consulta ao banco de dados. Os valores mais significativos de p-value são aqueles próximos de 0 (zero). Vide http://www.ncbi.nlm.nih.gov/Education/BLASTinfo/glossary2.html.

[10] **Qualidade PHRED** – ou ¨qualidade¨ da base, corresponde a um inteiro entre 0 e 99 e está associada à probabilidade de erro de leitura. Uma base com qualidade 40 indica que o erro é de 1 base em 10^4 (a qualidade é o expoente multiplicado por 10). Empiricamente, considera-se uma base como aceitável se tiver qualidade de, no mínimo, 20 (1 base incorreta em 100). Vide www.biotecnologia.com.br/bio/bio28/28_seque.pdf.

[11] **Rede Biofoco** – é a Rede de Bioinformática do Centro-Oeste, da qual fazem parte a Universidade Católica de Brasília, a Embrapa-Recursos Genéticos e a Universidade de Brasília, todas sediadas no Distrito Federal, Brasil.

[12] **Proteoma** – é o conjunto de todas as proteínas que intervêm nos processos biológicos de uma espécie. O principal objetivo de projeto proteômico é determinar a composição, estrutura, e funções das proteínas, ou seja, conhecer a função dos genes anteriormente descritos. Projetos genoma e proteoma estão altamente correlacionados, uma vez que a função de um gene está normalmente relacionada à proteína que ele codifica.Vide **[Pappas, 2002].**

[13] **Anotação** – uma combinação de comentários, notas, referências bibliográficas e citações em formato livre ou utilizando um vocabulário controlado, que juntos descrevem toda a informação experimental e inferida sobre um gene ou proteína. Anotações podem ser aplicadas para descrição de outros sistemas biológicos. Vide http://www.library.csi.cuny.edu/%7Edavis/Bioinfo_326/bioinfo_glossary.html.

[14] **Módulo de Anotação** – é um sistema que está sendo desenvolvido pelo colega André Barreto, da Universidade Católica de Brasília, como projeto final do curso *latu sensu* em Sistemas Orientados a Objetos do ano de 2002. O objetivo é capturar os dados de anotação de uma determinada seqüência.

[15] **Formato de seqüência** - É simplesmente a maneira que uma seqüência de aminoácido ou de DNA é escrita e gravada em um arquivo de computador. Diferentes programas esperam diferentes formatos. Por isso, para se obter o resultado esperado, é necessário conhecer os diversos formatos e sua estrutura básica. Segundo a literatura, os principais formatos de seqüências são: ClustalW, Codata, EMBL, GCG/MSF, GDE, Genebank, Fasta (Pearson), NBRF/PIR, Phylip, Swiss-Prot, Raw (Staden) e RSF. Vide http://www.molbiol.ox.ac.uk/help/sequenceformats.htm e http://www.ebi.ac.uk/help/formats_frame.html.

[16] **Alinhamento *pairwise*** – tipo de alinhamento que envolve apenas um par de seqüências.

[17] **Lista de preferências** – é uma lista que reúne os principais assuntos de interesse do pesquisador. Neste Documento, essa lista é também chamada de "*wish list*".

[18] **Bioinformática** - segundo o SNP Consortium (http://snp.cshl.org/help/glossary.shtml), é a ciência que usa avançadas técnicas computacionais para gerenciamento e análise de dados biológicos. A Bioinformática é particularmente importante como um auxílio à pesquisa genômica, a qual gera uma larga quantidade de dados complexos, envolvendo bilhões de blocos individuais de DNA e dezenas de milhares de genes.

[19] **Eucarioto** - segundo o dicionário Aurélio Século XXI, significa 1. organismo composto por uma ou mais células que possuem núcleo distinto, envolvido por membrana nuclear; eucarionte. Esse organismo é mais especializado que seu opositor **procarioto**, que é formado por uma única célula desprovida de membrana nuclear; procarionte.

[20] **Célula somática** - é qualquer célula do corpo, exceto as células reprodutoras. Vide http://www.ornl.gov/TechResources/Human_Genome/glossary/glossary.html.

[21] **Cadeias polinucleotídicas** - segundo o dicionário Aurélio Século XXI, significa 1. série de átomos ininterruptamente ligados entre si, em geral numa molécula orgânica; 2. polímero formado pela ligação de nucleotídeos.

[22] **Antiparalelo** - segundo o dicionário Aurélio Século XXI, significa 1. duas grandezas a que se podem atribuir direção e sentido, que têm a mesma direção mas sentidos opostos.

[23] **Púrica** - segundo o dicionário Aurélio Século XXI, significa 1. qualquer base derivada da purina e que entra na composição de inúmeras substâncias naturais.

[24] **Pirimidínica** - segundo o dicionário Aurélio Século XXI, significa 1. relativo à substância heterocíclica aromática, isômero da pirazina com dois átomos de nitrogênio separados por um de carbono, da qual se podem considerar derivadas três das bases nitrogenadas encontradas nos ácidos nucléicos, a citosina, a timina e a uracila [fórmula: C4H4N2].

[25] **Histona** - segundo o dicionário Aurélio Século XXI, significa 1. qualquer de certas enzimas que se ligam firmemente ao ácido desoxirribonucleico dos cromossomos de células eucarióticas.

[26] **Promotores** – regiões de DNA localizados fora da região codante dos genes, mas que definem o padrão de transcrição desses (os frames, vide ORF).

[27] **Satélites** – estrutura esférica na extremidade de um dos cromossomos. Vide http://intermega.com.br/biotemas/cromossomo_estrutura.htm. É diferente de DNA satélite, que envolve sequências repetidas (em tandem) agrupadas em um ou em alguns locais, intercaladas com sequências de cópia única ao longo do cromossomo. As famílias de DNA satélite variam quanto à localização no genoma, comprimento total da série em tandem, comprimrnto das unidades repetidas que constituem a série. Vide http://www.virtual.epm.br/cursos/genetica/htm/estru.htm.

[28] **DNA lixo** - ou *junk DNA* em inglês, corresponde aos íntrons e às seqüências repetitivas, muitas delas resultantes de: pareamento incorreto durante a replicação, ação da incorporação de vírus ou movimentação de transpósons (qualquer segmento móvel de ADN, capaz de mudar sua posição no genoma, e que pode ser empregado para neste introduzir material genético exógeno), micro e minissatélites.

[29] **Citosol** - sinônimo de citoplasma.

[30] **Proteínas globulares ou fibrosas** – enquanto as globulares são tipicamente solúveis em água, ligeiramente esféricas e bastante enoveladas, as fibrosas são insolúveis em água, fisicamente "tought" e constituídas de estruturas repetidas.

[31] **Sinapse** - segundo o dicionário Aurélio Século XXI, significa 1. (Histologia) conexão entre dois neurônios vizinhos, da qual há mais de um tipo, segundo as formações que fazem o contato entre essas células para que se propague o impulso nervoso de uma para outra.

[32] **Motivos** – ou *motifs* em inglês, é uma pequena região conservada (não alterada durante a evolução) na seqüência de uma proteína. Esses motivos (ou padrões) são, freqüentemente, partes altamente conservadas dos domínios.

[33] **Domínios** – uma discreta porção de uma proteína que se enovela independentemente do resto da seqüência e possui função própria.

[34] **ORF** – *Open Reading Frame* ou Região Aberta de Leitura, é uma seqüência contígua do mRNA e possui um códon de início (START CODON) e de terminação (STOP CODON). Dependendo de onde começa o processo de tradução, pode assumir 6 possibilidades (1ª. 2ª, ou 3ª. base da fita 1 ou 1ª. 2ª, ou 3ª. base da fita complementar). Seu tamanho é múltiplo de 3.

[35] **Éxon** - segundo o dicionário Aurélio Século XXI, significa 1. região do DNA de eucariotos que, após a transcrição, permanece no ácido ribonucléico. Essa seqüência propicia a síntese de proteína e é, por isso, chamada de região codante.

[36] **Subsérie** – ou *substring* em inglês, significa seqüência de caracteres coincidentes que forçosamente estão em posições adjacentes com relação àqueles da seqüência a que são comparados, por exemplo a subsérie a = opqab com relação a b = mnopqabc.

[37] **Árvore de sufixos** – ou *suffix tree* em inglês, é uma representação compacta de uma árvore de strings (trie) correspondente aos sufixos de uma dada string, onde todos os nós com um filho são mesclados com seus pais. Vide http://www.cc.ioc.ee/jus/gtglossary/gtglos_sp_sz.htm#suffix.

[38] **Primer** – é uma curta e preexistente cadeia polinucleotídica à qual novo DNA pode ser adicionado por DNA polymerase. Vide http://www.cstl.nist.gov/biotech/strbase/glossary.htm.

[39] **PCR** – *Polimerase Chain Reaction* ou Reação em Cadeia da Polimerase, é o método que permite a amplificação de qualquer região específica do DNA (um gene, uma região repetitiva etc) a partir do DNA genômico. Deste modo, pode-se, por exemplo, obter milhões de cópias do fragmento de DNA de 300 pb dentro do gene CFTR (cujo defeito leva à doença Fibrose Cística). Pode-se analisá-lo posteriormente com digestões de enzimas de restrição e técnicas de eletroforese para detecção da mutação. Vide http://www.icb.ufmg.br/~lbem/aulas/grad/evol/genpop.html.

[40] **Palíndromo** – segundo o dicionário Aurélio Século XXI, significa 1. frase ou palavra que, ou se leia da esquerda para a direita, ou da direita para a esquerda, tem o mesmo sentido, como, por exemplo, radar e ovo. Em Biologia Molecular, significa seqüência que apresenta correspondência exata com sua inversa, por exemplo, ATCGCCAT e TACCGCTA.

[41] **Subseqüência** – ou *subsequence* em inglês, significa seqüência de caracteres coincidentes que não necessariamente estão em posições contíguas com relação àqueles da seqüência a que são comparados, por exemplo a subseqüência a = nqab com relação a b = mnopqabc.

[42] **Similaridade** – segundo **[Meidanis e Setubal, 1997]**, significa uma medida de quão similares são as seqüências envolvidas.

[43] **Homologia** – segundo o dicionário Aurélio Século XXI, significa 1. semelhança de estrutura e de origem, em partes de organismos taxonomicamente diferentes.

[44] **Mielina** – segundo o dicionário Aurélio Século XXI, significa 1. substância lipóide que forma a bainha em torno de certos nervos.

[45] **Globina** – segundo o dicionário Aurélio Século XXI, significa 1. o componente protéico da hemoglobina. A hemoglobina é o transportador de oxigênio em hemácias de vertebrados. Sua molécula é um tetrâmero e cada subunidade compõem-se de uma cadeia polipeptídica, a globina, e um grupo prostético, o heme, um pigmento contendo ferro que se combina com o oxigênio e confere à molécula sua capacidade de transportar oxigênio (http://www.virtual.epm.br/cursos/genetica/htm/hemoglo.htm) .

[46] **Desigualdade triangular** – é um dos axiomas básicos das medidas de distância, interpretado da seguinte forma: a distância entre A e B é menor que a soma das distâncias de A a C e de C a B.

[47] **Citocromo c** – segundo o dicionário Aurélio Século XXI, significa 1. qualquer de certas proteínas que contêm heme e que participam de processos de oxirredução e transporte de elétrons, como ocorre, p. ex., na fotossíntese e na respiração.
[48] **Mioglobina** – segundo o dicionário Michaelis, significa 1. hemoglobina do músculo vermelho.
[49] **Hemoglobina** – segundo o dicionário Aurélio Século XXI, significa 1. pigmento existente na hemácia, formado por *heme* e globina, e entre cujas funções estão as de fixação do oxigênio atmosférico e sua transferência às células. Ainda segundo o dicionário Aurélio Século XXI, heme significa porfirina (grupos vermelho-escuros ou púrpura-escuros) que contém ferro e, unida à globina, constitui a hemoglobina, e que entra, também, na constituição de vários pigmentos respiratórios e de muitas células, tanto animais quanto vegetais.
[50] **Por acaso** – ou "*by chance*", por mera coincidência.
[51] **Gap –** significa lacuna, vácuo, intervalo ou distância em português.
[52] **Crossover desigual –** ou *unequal crossing-over* em inglês, ocorre durante a meiose e deve-se ao fato de os cromossomos homólogos se encontrarem desalinhados no momento do *crossover*. O desemparelhamento dos pares de genes em cada cromossomo homólogo leva a que, após o *crossover*, os cromossomos resultantes possuam duplicação e deleção de um ou mais genes. Vide http://eden.dei.uc.pt/~abs/asimoes.pdf.
[53] **Meiose –** segundo o dicionário Aurélio Século XXI, significa 1. processo de divisão pelo qual as células-filhas têm metade dos cromossomos da célula-mãe.
[54] **Retrovírus –** segundo o dicionário Aurélio Século XXI, significa 1. cada um de um grande grupo de vírus RNA no qual estão incluídos os leucovírus e os lentivírus.
[55] **Translocações –** segundo o dicionário Michaelis, significa 1. mudança de posição de um segmento de um cromossomo para outra parte do mesmo ou de outro cromossomo.
[56] **Sensitividade –** é a habilidade de um método em reconhecer seqüências distantemente relacionadas.
[57] **Mutação pontual aceita –** o termo, neste contexto, significa uma mutação que foi incorporada na proteína e passada aos seus descendentes. Então, ou a mutação não mudou a função da proteína ou a mudança na proteína foi benéfica ao organismo, ou, pelo menos, não se mostrou letal.
[58] **Expressões regulares –** segundo http://www.oreilly.com/catalog/regex/desc.html, são ferramentas para manipular segmentos de textos e dados que seguem um determinado padrão.
[59] **Pfam –** ou ***P****rotein* ***Fam****ily Database* (em inglês) de Alinhamentos e HMMs, segundo **[InterPro, 2001]**, é uma grande coleção de alinhamentos múltiplos de seqüências e modelos ocultos de Markov (HMM) envolvendo diversos domínios comuns de proteínas.
[60] **PRINTS –** ou *Protein Fingerprint Database* em inglês, segundo **[InterPro, 2001]**, é um compêndio de "impressões digitais" de proteínas. Uma impressão é um grupo de motivos conservados usados para caracterizar uma família de proteínas; e seu poder de diagnóstico é refinado por uma varredura iterativa de um composto de SWISS-PROT + SP-TrEMBL. Geralmente os motivos não se sobrepõem, mas estão separados ao longo da seqüência, embora possam ser contíguos em um espaço tridimensional. Essas impressões podem codificar o enovelamento e as funcionalidades de uma proteína mais flexivelmente que um simples motivo, pois a potência de seu diagnóstico completo deriva do contexto mútuo permitido pelos motivos vizinhos.
[61] **ProDom –** ou *The Protein Domain Database* em inglês, segundo **[InterPro, 2001]**, é um banco de dados de proteínas que consiste de uma compilação automática de domínios homólogos. As versões correntes são construídas utilizando um novo procedimento baseado nas buscas recursivas PSI-BLAST.
[62] **SMART –** ou *Simple Modular Architecture Research Tool* em inglês, segundo **[InterPro, 2001]**, é uma ferramenta simples de pesquisa de arquitetura modular que permite a identificação e anotação de domínios geneticamente móveis e a análise de arquiteturas de domínios.
[63] **TIGRFAMs –** ou *The Institute for Genomic Research Protein Families* em inglês, segundo **[InterPro, 2001]**, é uma coleção de famílias de proteínas, retratando curated alinhamentos múltiplos de seqüências, modelos ocultos de Markov (HMM) e anotação, que provê uma ferramenta para identificar proteínas funcionalmente relacionadas, baseando-se na homologia da seqüências.
[64] **Blocos –** segundo **[Gusfield, 1997]**, são pequenos intervalos contínuos em um alinhamento múltiplo sem lacunas de seqüências de aminoácidos.
[65] **Heurística** – segundo o dicionário Aurélio Século XXI, significa 1. do latim, 'arte de encontrar', 'descobrir'; 2. conjunto de regras e métodos que conduzem à descoberta, à invenção e à resolução de

problemas; 3. metodologia, ou algoritmo, usada para resolver problemas por métodos que, embora não rigorosos, geralmente refletem o conhecimento humano e permitem obter uma solução satisfatória.

[66] **Frameshift** – é uma mudança em como a informação em um gene é traduzida pela célula. Ocorre quando uma mutação de exclusão ou de inserção não é múltiplo de três pares de bases. Por exemplo, se uma exclusão tem como resultado a remoção de G da seqüência de mRNA "GAACCUA", os dois primeiros códons mudam de GAA e CCU para AAC and CUA, alterando, assim, a seqüência de aminoácidos especificadas pelo mRNA.

[67] **Seletividade** – é a habilidade de um método em evitar aquelas seqüências não-relacionadas, mesmo que apresentem altos escores de similaridade.

[68] **REPBASE** – segundo Geospiza Inc., é um banco de dados de elementos repetitivos de seqüências de DNA encontrados em uma variedade de organismos eucarióticos, incluindo mamíferos, peixes, insetos, nematóides e plantas.

[69] **EST** – ou *Expressed Sequence Tags* em inglês, segundo http://www.library.csi.cuny.edu/%7Edavis/Bioinfo_326/bioinfo_glossary.html#e, significa uma pequena seqüência de um gene expresso que pode ser amplificado por PCR. ESTs atuam como marcadores físicos para clonagem e seqüenciamento completo dos cDNAs de genes expressos. Tipicamente identificados purificando mRNAs, convertendo para cDNAs, e, então, seqüenciando uma porção dos cDNAs.

[70] **STS** – ou *Sequence Tagged Site* em inglês, segundo http://www.library.csi.cuny.edu/%7Edavis/Bioinfo_326/bioinfo_glossary.html#s, significa uma única seqüência de localização cromossômica conhecida que pode ser amplificada por PCR. STSs atuam como marcadores físicos para mapeamento genômico e clonagem.

[71] **GSS** – ou *Genome Survey Sequence* em inglês, segundo http://www.genomicglossaries.com/content/lifesciences_databasesdirectory.asp , é uma divisão do GenBank (http://www.ncbi.nlm.nih.gov/dbGSS/) similar à divisão de EST, exceto pelo fato de que suas seqüências são genômicas em origem, e não cDNA (mRNA).

[72] **HTGS** – ou *High Throughput Genomic Sequences* em inglês, segundo http://www.genomicglossaries.com/content/lifesciences_databasesdirectory.asp, é uma divisão do GenBank (http://www.ncbi.nlm.nih.gov/HTGS/) criada para acomodar seqüências de DNA não finalizado geradas pelos centros de seqüenciamento high-throughput.

[73] **Matriz de escore unitária** – segundo http://www.ncbi.nlm.nih.gov/Education/BLASTinfo/glossary2.html, esta matriz é também conhecida como matriz identidade. É um sistema de escore no qual apenas caracteres idênticos recebem um escore positivo.

[74] **Paradigma ou Máquina de Mealy** – segundo http://www.inf.ufrgs.br/pos/SemanaAcademica/Semana99/juliohapm/juliohapm.html, a Máquina de Mealy é um autômato finito modificado de forma a gerar uma palavra de saída para cada transição.

[75] **SP (soma dos pares)** – ou sum-of-pairs em inglês, significa a soma dos escores de todos os pares de símbolos em cada coluna, sendo as seqüências tomadas duas a duas **[Meidanis e Setubal, 1997]**.

[76] **Mutagêneses** – segundo o dicionário Aurélio Século XXI, significa 1. Genét. Processo que dá origem às mutações. 2. Citol. Produção de mutações em uma população de células.

[77] **Modelos Ocultos de Markov (HMM)** – são modelos estatísticos do consenso de seqüência de uma família homóloga. Vide http://smart.embl-heidelberg.de/help/smart_glossary.shtml.

[78] **Marcadores** – são genes de localização conhecida em um cromossomo e fenótipo, os quais são usados como um ponto de referência no mapeamento de outro loco (loci). Vide http://www.cstl.nist.gov/biotech/strbase/glossary.htm.

[79] **Táxon** - é um conjunto de organismos biológicos caracterizado por propriedades compartilhadas por todos os organismos, segundo http://www.ime.usp.br/~rsouza/ic/node3.html.

[80] **Parafilético** – um grupo de organismos que descendem de um ancestral comum, mas nem todos os organismos derivados são incluídos.

[81] **Polifilético** – um grupo no qual são colocados organismos que não compartilham um ancestral imediato comum, enquanto exclui outros membros que poderiam ligá-los.

[82] **Ergonomia** - estudo que visa a tornar as relações entre o homem e a máquina harmoniosas, impedindo o surgimento de problemas relacionadas à postura. (www.babylon.com)

[83] **Herança** – é um mecanismo que permite que características comuns a diversas classes sejam fatoradas em uma classe base, ou superclasse. A partir de uma classe base, outras classes podem ser especificadas. Cada classe derivada ou subclasse apresenta as características (estrutura e métodos) da classe base e acrescenta o que lhe for definido de particularidade. Vide http://www.dca.fee.unicamp.br/courses/PooJava/heranca/.

[84] **Subclasse** – é a classe derivada de uma superclasse.

[85] **Encapsulamento** – é a separação dos aspectos externos de um objeto dos seus detalhes internos de implementação. Também chamado de ocultamento de dados. Vide http://www.ppgia.pucpr.br/~alcides/Teaching/2000/dawjaw/Encapsulamento.ppt.

[86] **Polimorfismo** – é o princípio pelo qual duas ou mais classes derivadas de uma mesma superclasse podem invocar métodos que têm a mesma identificação (assinatura) mas comportamentos distintos, especializados para cada classe derivada, usando para tanto uma referência a um objeto do tipo da superclasse, a chamada sobreposição. A decisão sobre qual o método que deve ser selecionado, de acordo com o tipo da classe derivada, é tomada em tempo de execução, através do mecanismo de ligação tardia. Vide http://www.dca.fee.unicamp.br/courses/PooJava/polimorf/index.html. Há também o polimorfismo chamado sobrecarga: há especialização de método dentro da própria classe e não envolve herança.

[87] **Vias metabólicas** – o metabolismo celular é um conjunto altamente organizado e complexo de reações bioquímicas, catalizadas e reguladas por enzimas. Essas reações se organizam em seqüências enzimáticas denominadas VIAS METABÓLICAS, as quais podem ser anabólicas (ocorrem com utilização de energia e são não-espontâneas) e catabólicas (ocorrem sempre com liberação de energia e são espontâneas). Vide http://members.tripod.com/medworks/Bioquimica/metabol.html.

[88] **Genealogia** – segundo o dicionário Aurélio Século XXI, significa 1. conjunto de descendentes dum indivíduo; 2. estudo da origem e formação do indivíduo ou da espécie.

[89] **COG** – *Clusters of Orthologous Groups* ou Agrupamentos de Grupos Ortólogos de proteínas, foi delineado pela comparação de sequências de proteínas de 43 genomas completos, representando 30 principais linhagens filogenéticas. Cada COG consiste de proteínas individuais ou grupos de parálogos de, no mínimo, 3 linhagens e, assim, corresponde a um domínio conservado. Proteínas de 2 genomas eucariotos (Drosophila melanogaster e Caenorhabditis elegans) foram designadas ao COG e podem ser visualisadas de cada página individual COG.Vide http://www.ncbi.nlm.nih.gov/COG/. Ou seja, bancos de genes classificados de acordo com sua história evolutiva.Vide http://www.icb.ufmg.br/~lbem/aulas/grad/tge/bioinfo/bioinfogenomica.pdf.